



# Combogeneraattori

WWW-sivusto trikkauksen harrastajille

Juhani Viitanen

Opinnäytetyö  
Toukokuu 2013  
Tietojenkäsittelyn koulutus-  
ohjelma  
Ohjelmistotuotanto

## TIIVISTELMÄ

Tampereen ammattikorkeakoulu  
Tietojenkäsittelyn koulutusohjelma  
Ohjelmistotuotanto

VIITANEN, JUHANI:  
Combogeneraattori  
WWW-sivusto trikkauksen harrastajille

Opinnäytetyö 65 sivua  
Toukokuu 2013

---

Opinnäytetyön aihe oli triikkaajille suunnatun WWW-sivuston toteuttaminen. Sivuston tavoitteena oli antaa triikkaajille uudenlainen työkalu uusien liikesarjojen luomiseen. Tarkoituksena oli toteuttaa liikesarjageneraattori ja rakentaa sen ympärille sivusto, jossa käyttäjät pystyvät generoimaan liikesarjoja sekä muokkaamaan ja tallentamaan niitä.

Opinnäytetyössä käsitellään käyttäjäkeskeisiä menetelmiä WWW-sivuston suunnittelun tukena. Työssä esitellään käytetyt tekniikat ja ohjelmistokehykset. Työssä käytettiin erilaisia ohjelmistokehyksiä ja kirjastoja, kuten Yii PHP-frameworkia, jQuerya ja Bootstrapia. Lisäksi työssä valotetaan sivuston rakentamista ja toteuttamiseen liittyviä eri vaiheita.

Sivuston liikesarjageneraattorista tuli hyvä ja juuri niin toimiva kun aluksi oli suunnittelukin. Työn edetessä lisättiin kuitenkin uusia ominaisuuksia sitä mukaa, kun saatiin vanhoja valmiiksi. Tästä syystä sivustoa ei julkaista suurelle yleisölle vielä opinnäytetyön tekemisen aikaan, vaan sivuston kehittäminen jatkuu opinnäytetyön palauttamisen jälkeen.

## **ABSTRACT**

Tampereen ammattikorkeakoulu  
Tampere University of Applied Sciences  
Business Information Systems  
Option of Software Engineering

VIITANEN, JUHANI:  
Combo Generator  
Website for Tricking Enthusiasts

Bachelor's thesis 65 pages  
May 2013

---

The subject of the thesis was a combo generator website for tricking enthusiasts. The aim of the website was to provide a new kind of tool for tricking enthusiasts for creating new kind of combos. The goal was to implement the generator and build a website around it. In the website users could edit and save generated combos.

This thesis deals with user-centered methods of web-site design. The thesis presents the techniques and software frameworks. A variety of frameworks and libraries was used, such as Yii PHP-framework, jQuery and Bootstrap. The thesis has also more information about construction and implementation of the different stages of building the website.

The generator built for the website was good, and just as effective as was initially planned. As the work progressed, new features were added when old ones started working. For this reason, the site is not published to the audience at the time of this thesis, but the site will be further developed after the submission of the thesis.

---

Key words: www, yii, php, user-centered design, tricking

## SISÄLLYS

1	JOHDANTO.....	3
2	SIVUSTON SUUNNITTELU .....	5
2.1	Tavoitteet ja tarkoitus .....	5
2.2	Käyttäjäkeskeinen suunnittelu .....	5
2.2.1	Käyttäjien vaatimukset ja tarpeet .....	7
2.2.2	Persoonat .....	9
2.2.3	Käyttötarinat.....	10
2.3	Käyttöliittymän suunnittelu .....	11
2.3.1	Sivuston rakenne .....	13
3	KÄYTETYT TEKNIIKAT .....	15
3.1	WWW-tekniikat.....	15
3.1.1	HTML .....	15
3.1.2	CSS.....	16
3.1.3	PHP .....	17
3.1.4	JavaScript .....	17
3.1.5	MySQL.....	18
3.2	Ohjelmistokehykset .....	18
3.2.1	Yii.....	19
3.2.2	jQuery.....	20
3.2.3	Bootstrap .....	20
3.2.4	LESS .....	21
4	SIVUSTON TOTEUTTAMINEN .....	22
4.1	Yii-projektin perusrakenne ja toiminta .....	22
4.2	Alkutoimenpiteet .....	24
4.3	Trikkaukseen liittyvät asiat .....	27
4.3.1	Liikkeet ja liikesarjat.....	27
4.3.2	Liikelistat .....	32
4.3.3	Liikesarjalistat .....	33
4.3.4	Tietokanta.....	33
4.3.5	Koodin generointi Giix-laajennuksella .....	35
4.3.6	Generaattori.....	39
4.4	Käyttäjien hallinta.....	56
4.5	Sivuston testaaminen .....	59
5	VALMIIN TYÖN ARVIOINTI.....	61
6	SIVUSTON JATKOKEHITYS .....	62

7 LOPPUSANAT .....	64
LÄHTEET .....	65

## 1 JOHDANTO

Halusin ehdottomasti tehdä opinnäytetyökseni jotain WWW-sivustoihin liittyvää, koska se aihe on minua koulussa eniten kiinnostanut ja haluaisin tehdä työkseni WWW-kehitystöitä. Opinnäytetyötäni toteutin WWW-sivuston, joka auttaa trikkauksen harrastajia uusien liikesarjojen luomisessa. Tässä opinnäytetyöraportissa käyn läpi vaiheittain sivuston toteuttamisen aina suunnittelusta valmiiseen sivustoon asti.

Trikkkaus (*tricking*) on vauhdikas urheilulaji, joka koostuu potkuista (*kicks*), kierteistä (*twists*) ja volteista (*flips*). Trikkkauksessa tehdään näyttäviä liikkeitä sekä näiden yhdistelmiä eli liikesarjoja. Trikkkaukseen on otettu vaikutteita mm. Capoeirasta, Tae Know Dosta, Karatesta, Wushusta, Voimistelusta ja Breakdancesta. Trikkkaus on lähtöisin 1990-luvun Yhdysvaltojen urheilukaratepiireistä, kun harrastajat halusivat liikesarjoihinsa enemmän näyttävyyttä lisäämällä niihin näyttäviä potkuja, kierteitä sekä voltteja. Trikkkaus on levinnyt internetin kautta kaikkialle maailmaan.

Trikkkauksessa on tarkoituksena tehdä mahdollisimman näyttäviä liikkeitä, sekä näiden yhdistelmiä eli liikesarjoja. Mahdollisia liikkeitä on kymmeniä, ja niitä yhdistelemällä saadaan tuhansia eri liikesarjoja. Tästä syystä moni lajin harrastaja pitää omia vihkoja, joihin kirjoitetaan liikesarjoja muistiin, jotta niitä muistaa harjoitella treeneissä. Itsekin lajia harrastavana ajattelin, että olisi hienoa, jos voisi arpoa itselleen täysin uusia liikesarjoja. TAMKissa dynaamista WWW-ohjelmointia opiskelleena uskoin osaavani toteuttaa generaattorin, joka arpoo uusia mielenkiintoisia ja toteutettavissa olevia liikesarjoja. Mietin asiaa ja keskustelin siitä muiden lajin harrastajien kanssa. Palaute oli myönteistä. Päätin, että toteutan WWW-sivuston, jossa on liikesarjoja luova generaattori ja virtuaaliset ”vihkot”, joihin näitä liikesarjoja voisi sitten tallentaa. Jotta saisin sivuston joskus valmiiksi, halusin tehdä siitä opinnäytetyöni aiheen. Sivustoa suunniteltaessa ideoita tuli niin paljon, että jouduin rajaamaan niistä suuren osan opinnäytetyöni käytännöntoteutuksen ulkopuolelle sillä tiesin, että en millään kerkeisi toteuttamaan kaikkea. Toivon, että mahdollisimman moni triikkaaja löytää sivuston ja kokee sivuston hyödylliseksi.

Tämän opinnäytetyön tavoite on kertoa lukijalle, mitä asioita otin huomioon, ja kuinka toteutin combogeneraattori sivuston. Pyrin kartoittamaan nykyaikaisia tapoja lähestyä

WWW-sivusto projektia. Opinnäytetyössäni käyn läpi kaikki vaiheet suunnittelusta, tekniikoiden valintaan ja itse sivuston toteutukseen.

Opinnäytetyön tarkoituksena on kertoa, mitä tein sivuston suunnittelusta aina valmiiseen sivustoon asti. Tämän opinnäytetyön ei kuitenkaan ole tarkoitus olla opas WWW-sivustojen tekemiseen. Toki työstäni voi ottaa mallia ja löytää hyviä vinkkejä etenkin hyödyntämieni tekniikoiden käyttöön.

Käsittelen tässä opinnäytetyössä ensin sivuston suunnitteluun liittyvät asiat. Käyn läpi käyttäjakeskeisensuunnittelun peruseriaatteita ja tekniikoita. Suunnittelun jälkeen esittelen kaikki käyttämäni WWW-tekniikat ja ohjelmistokehykset, sekä kerron, miten näitä käytin. Kun suunnittelut ja tekniikat ovat selvillä, on aika käydä läpi sivuston toteutus. Kerron vaihe vaiheelta, miten sivusto valmistui, ja kuinka se toimii.

Lähteitä WWW-ohjelmointiin ja WWW-sivuihin liittyen on löytynyt erittäin paljon, joten oli varaa valita parhaat päältä. Olen pyrkinyt valitsemaan käyttämäni lähteet luotettavilta julkaisijoilta. Lähteitä valitessani olen kiinnittänyt huomiota siihen, kuinka tuore lähde on. WWW ja sen kehitys muuttuu kovaa tahtia ja vanhemmat kirjat ovat jo lähes antiikkisen tuntuisia.

## 2 SIVUSTON SUUNNITTELU

### 2.1 Tavoitteet ja tarkoitus

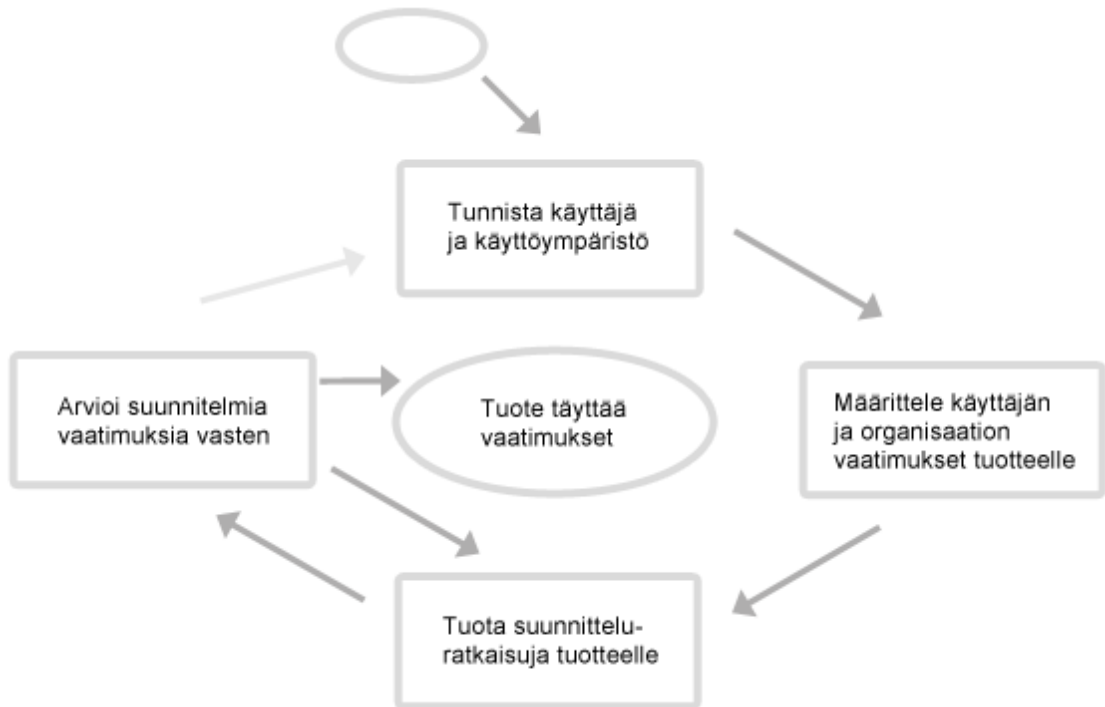
Sivuston tavoitteena on antaa triikkaajille työkalut helpottamaan uusien liikesarjojen keksimistä. Toivon, että sivusto innostaa triikkaajia tekemään monipuolisempia liikesarjoja ja opettelemaan niitäkin liikesarjoja, jotka ovat oman mukavuusalueen ulkopuolella. Tavoitteena sivustoa suunniteltaessa on tutustua käyttäjäkeskeisiin menetelmiin.

Tarkoituksena on toteuttaa WWW-sivusto, joka mahdollistaa edellä mainitut tavoitteet. Sivustolle toteutetaan generaattori, joka arpoo käyttäjän antamien tietojen perusteella toteutettavissa olevia liikesarjoja. Sivuston suunnittelussa on tarkoitus käyttää käyttäjäkeskeisiä menetelmiä.

### 2.2 Käyttäjäkeskeinen suunnittelu

Käyttäjäkeskeinen suunnittelu lähtee liikkeelle käyttäjien tunnistamisesta. Keitä käyttäjät ovat, miten ja missä ympäristössä käyttäjät toimivat, sekä mihin he tuotetta käyttäisivät. Näiden tietojen avulla tehdään tuote ja testataan, tuliko siitä hyvä. Suunnitteluratkaisujen perustana ovat käyttäjätutkimukset, sekä auki kirjoitetut liiketoiminnalliset tavoitteet. Tämä on käyttäjäkeskeisen suunnittelun perusidea, joka toistuu kaikissa käyttäjäkeskeisissä menetelmissä ja on kuvattu myös ISO 13407-standardissa. (Kuva 1) (Sinkkonen, Nuutila & Seppo 2009, 33)





Kuva 1 ISO 13407-standardi Sinkkosen korjaama versio (Sinkkonen ym. 2009, 34)

Käyttäjakeskeisiä menetelmiä on useampia. Käytin niistä kuitenkin vain niitä, joiden koin olevan minulle hyödyllisiä sivuston suunnittelussa. Esittelen seuraavaksi käyttämäni tekniikat. Näiden lisäksi olin aktiivisesti vuorovaikutuksessa tulevien käyttäjien kanssa. Esittelyn jälkeen kerron seuraavissa alaluvuissa, kuinka käytin näitä tekniikoita tässä projektissa.

### **Persoonat**

Persoonat ovat käyttäjätutkimuksessa tai aiemmin löydettyjen käyttäjien eräänlaisia tiivistelmiä. Nämä persoonat edustavat käyttäjiä palvelun suunnittelun aikana. Persoonat auttavat lähestymään sivuston suunnittelua enemmän käyttäjän näkökulmasta. (Sinkkonen ym. 2009, 33)

### **Toimintatarinat ja käyttötarinat**

Toimintatarinoissa käydään läpi käyttäjän toiminta ennen uutta palvelua. Käyttötarinoissa taas puolestaan kerrotaan, kuinka käyttäjä toimii uudessa palvelussa. (Sinkkonen ym. 2009, 33)

Tarinoita on helpompia keksiä, kun ne tehdään persoonia hyväksi käyttäen. Kuvittelemalla, miten persoona käyttäytyy, ja mitä ominaisuuksia hänellä on. Näin saadaan laajempi ja tarkempi kuva siitä, kuinka palvelu sopeutuu suunnitellulle käyttäjäkunnalle.

Käyttötarinoissa käydään läpi jokainen vaihe, mitä käyttäjä palvelussa tekee. Tämä usein jo paljastaa totuuden, jos palvelussa on käytettävyysoongelmia. Asiat tulevat käsiteltyä paljon järjestelmällisemmin ja useat pienet yksityiskohdat tulevat paremmin esille, kuin normaalissa suunnittelussa. Kun tarinat kirjoitetaan suunnitteluvaiheessa, voidaan useat käytettävyyvirheet korjata jo ennen kuin ensimmäistäkään riviä on aloitettu ohjelmoimaan. Mitä aikaisemmin virheet löydetään, sitä halvemmaksi niiden korjaaminen tulee. (Sinkkonen ym. 2009, 124–125)

### **2.2.1 Käyttäjien vaatimukset ja tarpeet**

Lähestyin käyttäjien tarpeita haastattelun ja pienimuotoisen vaatimusmäärittelyn muodossa. Käyttäjien haastattelemisen lisäksi poimin joitain kohtia Sinkkosen kirjassaan määrittelemästä listasta, joka kertoo vaatimusmäärittelystä. Käyttäjien omien toiveiden lisäksi mietin vaatimuksia, jotka tekevät palvelusta turvallisen ja mieluisan käyttää. Vaatimusmäärittelyssä määrittelin, mitä ominaisuuksia sivustolla täytyy vähintään olla, ja mitä asioita käyttäjien pitää pystyä tekemään. Sain itselleni käsityksen siitä, mikä olisi tärkeintä toteuttaa, jotta käyttäjien tavoite sivustolla täytyisi. Kaikki muu olisi vain plussaa ja voi odottaa myöhempään kunnes päätoiminnallisuus on saatu valmiiksi.

Aloitin suunnittelun lajin harrastajia haastattelemalla. Haastattelut olivat hyvin vapaa-muotoisia. Kyselin, mitä ominaisuuksia sivustolle tarvitaan, ja kerroin avoimesti, mitä olin itse suunnitellut. Sivuston toteutuksen edetessä julkaisin uusia versioita, joita tulevat käyttäjät saivat käyttää ja kertoa mielipiteensä. Pyrin ottamaan kaikki mielipiteet huomioon ja toteuttamaan niistä fiksuimmat ideat ajan ja taitojeni niin salliessa.

Haastattelemistani lajin harrastajista moni oli innoissaan kertomistani suunnitelmista ja kannustavat tekemään sivuston pian valmiiksi. Yleisimmät kysymykset olivat; toimiiko se älypuhelimella, saako generaattoriin lisätä omia temppuja ja pystyykö generoitavien liikesarjojen sisältöön vaikuttamaan itse? Nämä haastattelut ja niistä esille tulleen toi-

veet ohjasivat suunnittelua ja toteutusta enemmän käyttäjien toivomaan suuntaan. Mielestäni tämä tapa suunnitella sivustoa oli erittäin toimiva, sillä olin kokoajan suorassa vuorovaikutuksessa loppukäyttäjien kanssa.

Vaatusmäärittelyn on tarkoitus auttaa ymmärtämään, mitä palvelussa pitää olla. Vaatusmäärittelyn voi jakaa useampiin osiin, mikä helpottaa vaatimusten tunnistamista. Kun tiedetään, mitä palvelulta vaaditaan, osataan se rakentaa enemmän halutun kaltaiseksi. (Sinkkonen ym. 2009, 49)

### **Toiminnalliset vaatimukset**

Mitä sivustolla voi tehdä? Tärkeintä on tietenkin liikesarjojen generointi generaattoria hyväksi käyttäen. Käyttäjän täytyy voida myös tallentaa haluamansa liikesarjat myöhempää käyttöä varten. Näiden lisäksi käyttäjät voivat hakea jo aikaisemmin tallennettuja liikesarjoja tietokannasta tai kirjoittaa täysin omia liikesarjoja, jotka käyttäjä voi sitten tallentaa muiden liikesarjojen sekaan.

Generaattori pitää olla muokattavissa juuri itselle sopivaksi. Eri triikkaajat osaavat eri liikkeitä, joten omien liikkeiden määrittäminen yhdessä liikesarjan pituuden kanssa ovat tärkeitä ominaisuuksia. Generaattorista ei myöskään voi löytyä heti kaikkia tarpeellisia liikkeitä, joten käyttäjille pitää antaa mahdollisuus omien liikkeiden lisäämiseen.

Sivustolle pitää pystyä rekisteröitymään sekä kirjautumaan sisään, jotta omien tietojen lisääminen ja säilyttäminen on mahdollista.

### **Toimintaympäristövaatimukset**

Generaattoria tullaan luultavasti käyttämään paljon treenien aikana, joten sivuston pitää toimia myös mobiililaitteilla, kuten älypuhelimet ja tabletit. Erikokoiset näytöt pienistä puhelimen näytöistä isoihin työpöytäkoneiden näyttöihin tulee olla tuettuna. Sivusto pitää olla käytettävissä kaikilla moderneilla WWW-selaimilla.

### **Käyttäjävaatimukset**

Combogeneraattorin käyttäjältä vaaditaan tuntemusta triikkausliikkeistä. Sivusto ei ainaakaan aluksi tarjoa tietoa erilaisista liikkeistä tai liikesarjoista. Usein aloittelevat triikkaa-

jat oppivat liikkeiden nimet nopeasti toisten triikkaajien kautta tai netistä löytyvistä tietolähteistä kuten videoista ja keskustelufoorumeilta.

### **Käytettävyysvaatimukset**

Sivuston pitää olla nopea ja helppo käyttää eri laitteilla. Ei liian monimutkaisia ja liikaa klikkauksia vaativia navigointiratkaisuja. Sivustolta löytyy kattavat ohjeet käyttäjille. Ohjeet ovat erittäin tärkeässä osassa, jotta käyttäjät ymmärtävät, kuinka generaattori toimii, ja saavat siitä paljon enemmän irti, kun osaavat käyttää sitä oikein. Tavoitteenani on saada sivustosta yksinkertainen ja tehokas.

### **Turvallisuusvaatimukset**

Käyttäjien tärkein ja kallein asia on heidän salasanansa. Sivuston pitää olla turvallinen, jotta käyttäjän salasana ei joudu väärin käsiin esimerkiksi tietomurron yhteydessä.

## **2.2.2 Persoonat**

Ensin tunnistetaan käyttäjät käyttäjätutkimuksen avulla ja sen pohjalta voidaan luoda käyttäjäprofiilit ja käyttäjäryhmät. Näistä eri käyttäjäryhmistä voidaan vielä tiivistää jokainen ryhmä omaksi persoonakseen. Näin saadaan pieni joukko kuvitteellisia käyttäjiä, joille sivustoa voidaan suunnitella. Tämä helpottaa suunnittelua, kun käyttäjät ovat konkreettisia hahmoja sen sijaan, että suunniteltaisiin vain abstraktille käyttäjäjoukolle. (Sinkkonen ym. 2009, 35)

En suorittanut varsinaista käyttäjätutkimusta selvittääkseni, millaisia tulevat käyttäjät voisivat olla. Olen harrastanut lajia jo lähes kymmenen vuotta ja koen, että minulla on jo hyvä käsitys siitä, millaiselle käyttäjäryhmälle sivustoa suunnittelen. Kehittelin persoonat oman kokemukseni tuomien tietojen avulla. Käyttäjät ovat noin 12–30-vuotiaita miehiä. Myös naisia mahtuu joukkoon muutama, mutta he ovat selvä vähemmistö. Käyttäjät ovat harrastaneet triikkausta 0-10 vuotta ja taidot vaihtelevat hyvin paljon henkilöiden välillä. Triikkauksen aloitusikä vaihtelee myös täysin henkilöstä riippuen.

Persoonat voi kirjoittaa täysin vapaamuotoisesti. Persoonalle keksitään nimi, joka voi olla myös persoonan jotain ominaisuutta kuvaava. Esimerkiksi nimi voisi olla Antti

Aloittelija. Nimen lisäksi persoonalle keksitään muut henkilötiedot. Näitä voisi olla vaikka ikä, asuinpaikka, perhe, harrastukset. Näitä tietoja käyttäen kirjoitetaan kuvaus persoonasta. (Sinkkonen ym. 2009, 127)

Persoonalle kirjoitetaan myös muita tärkeitä tietoja, jotka liittyvät sivuston käyttämiseen. Näitä ovat mm. tietotekninen osaaminen ja käytössä olevat laitteet, käyttöympäristö, eli missä persoona käyttää sivustoa, ja persoonan tavoitteet sivustolla. (Sinkkonen ym. 2009, 128)

Jaoin käyttäjäryhmät aloittelijoihin, muutaman vuoden harrastaneisiin ja pitkään lajia harrastaneisiin. Näiden pohjalta kirjoitin kolme persoonaa, joita käytin suunnitteluni apuna.

### **2.2.3 Käyttötarinat**

Kun käyttäjäprofiilien pohjalta on tehty persoonat, voidaan vielä kehitellä sopivat käyttötarinat, joissa käydään läpi tilanteita vaihe vaiheelta, kuten käyttäjä palvelua käyttäisi. Ennen sivuston toteuttamista käydään toimintatarinoissa läpi se, miten käyttäjät toimivat aikaisemmin. Käyttötarinoissa käydään läpi toiminta uudella sivustolla. Tarinoissa kerrotaan aina yhdestä persoonasta kerrallaan, mutta persoonasta voi olla useita tarinoita eri tilanteista. (Sinkkonen ym. 2009, 135)

Varsinaista toimintatarinaa en kirjoittanut, koska vastaavanlaista sivustoa ei ole aikaisemmin ollut. Ennen combogeneraattoria uusia liikesarjoja keksittiin itsenäisesti ja kopioitiin muilta. Combogeneraattorin tarkoitus ei kuitenkaan ole syrjäyttää mitään aikaisempaa tapaa luoda uusia liikesarjoja, vaan tuoda täysin uusi tapa näiden muiden tapojen tueksi. Triikkaajien pitää ehdottomasti myös itse miettiä uusien liikesarjojen kehittelyä, mutta generaattorista voi saada sellaisia ideoita, joita ei itse ole koskaan tullut ajatelleeksi. Triikkaaja voi ottaa generaattorin käyttöön muiden liikesarjojen luonti tapojen lisäksi.

Kirjoitin useita käyttötarinoita. Tein uuden aina, kun olin päässyt suunnittelussa ja toteutuksessa pidemmälle, ja aloin epäilemään sivuston käytettävyyttä. Joka kerralla käyt-

tötarinasta oli apua. Tarinaa kirjoittaessa huomasi heti, jos jotain olennaista puuttui suunnitelmista. Tarinoiden kirjoittamisessa tulee mietittyä huomattavasti tarkemmin, mitä käyttäjän pitää voida tehdä sivustolla, ja mitä hän todennäköisesti haluaisi tehdä. Käyttötarinoita kirjoittaessani pyrin käymään yksityiskohtaisesti vaihe vaiheelta, mitä persoona sivustolla tekee ottaen huomioon persoonan erityispiirteet. Tämä muokkaa jokaisen persoonan tarinaa aina erilaiseksi ja saan käsiteltyä sivuston suunnittelua tarkemmin ja laajemmin.

### **2.3 Käyttöliittymän suunnittelu**

Helppokäyttöisen käyttöliittymän saavuttamiseksi voidaan listata tiettyjä periaatteita, joita noudattamalla käyttöliittymästä saadaan helpommin käytettävä. (Sinkkonen ym. 2009, 35)

Käyttöliittymän tulisi tukea käyttäjien toimintatapoja. Ensin selvitetään, kuinka käyttäjät suorittavat tehtävänsä normaalisti, ja näitä tietoja sovelletaan käyttöliittymän suunnitteluun, jotta käyttöliittymä tukee käyttäjien tapoja tehdä tehtäviään. (Sinkkonen ym. 2009, 35)

Navigointi on erittäin tärkeä osa sivustoa. Navigointi olisi syytä saada mahdollisimman selkeäksi ja tehokkaaksi. Navigointiin liittyen käyttäjän pitää tietää aina, millä sivulla hän on, ja missä tilassa sivusto kulloinkin on. Käyttäjille pitää antaa vaihtoehtoja liikkua sivustolla monipuolisesti. Käyttöliittymään ei saa syntyä umpikujia. (Sinkkonen ym. 2009, 36)

Yhdenmukainen käyttöliittymä on selkeämpi ja helpompi käyttää. Eri sivut voivat toki vaihdella asettelultaan tarpeen mukaan, mutta sivuston pitää noudattaa yhtenäistä kaavaa. Visuaalisen ilmeen tulisi pysyä yhtenäisenä sivuston jokaisella sivulla, ja tärkeiden elementtien, kuten navigoinnin löytyä aina helposti. (Sinkkonen ym. 2009, 36)

Sivustolle ei kannata lisätä toimintoja, joita ei käytetä. Käyttäjätutkimuksen ja testaamisen avulla selvitetään, mitä asioita käyttäjät tarvitsevat, ja käyttöliittymään sisällytetään nämä asiat. (Sinkkonen ym. 2009, 36)

Käyttöliittymän tulee opastaa käyttäjää sen verran kuin on tarpeen. (Sinkkonen ym. 2009, 36) Combogeneraattorissa ainakin generaattorin toimintatavat täytyy olla selitettynä käyttäjille. Tämä on tarpeen, jos käyttäjä haluaa muokata generaattoria enemmän omiin tarpeisiinsa. Myös sivuston muista toiminnoista pitää löytyä opastus käyttäjälle.

### **Eri päätelaitteet**

Sivustoa pitää pystyä käyttämään vaivattomasti niin älypuhelimilla kuin työpöytä-koneillakin. Laitteet, näyttöjen koko ja ympäristö siis vaihtelevat paljon. Käyttöliittymän suunnittelussa pitää ottaa huomioon näytön koko ja ympäristö, missä sivustoa käytetään. Combogeneraattoria tullaan varmasti käyttämään paljon mobiililaitteilla harjoitusten yhteydessä. On siis syytä ottaa kaikki laitteet huomioon heti sivuston suunnittelun alkuvaiheessa.

Hyvä lähestymistapa sivuston suunnitteluun on ”mobile first” -periaate. Tällä tahdotaan korostaa sitä, että sivuston tulisi toimia myös mobiililaitteilla. Kun suunnitellaan ensin pienimmälle näytölle käyttöliittymän asetteluja ja toimintoja joudutaan miettimään tarkemmin, ja usein tuloksena on yksinkertaisempi ja helppokäyttöisempi käyttöliittymä, kun sivu sisältää vain tärkeimmät asiat. (Wroblewski 2011, 19-22)

Ulkoasu ei siis voi olla sama isoille työpöytänäytöille ja pienille älypuhelimien näytöille. Päätin ratkaista asian käyttämällä responsiivista, eli näytön kokoon mukautuvaa, ulkoasua. Elementtien järjestys muuttuu ja osa elementeistä voidaan myös piilottaa tai tuoda näkyviin näytön koosta riippuen. Näin sama käyttöliittymä voidaan muotoilla monelle eri laitteelle.

### **Iterointi**

Käyttöliittymä pitäisi käyttäjälähtöisten periaatteiden mukaan suunnitella ja testata ensimmäiseksi ennen kuin sivustoa lähdetään tekemään. Minun tilanteessani näin ei käynyt. Halusin ensin saada selväksi, mitä ominaisuuksia pystyn toteuttamaan, ja kuinka hyvin. Käyttöliittymässä olisi kuitenkin ollut jotain niin hienoa, että sen toteuttaminen olisi voinut olla liian vaikeata ja siitä syystä joutuisin sitä muuttamaan. Siispä päätin ensin rankentaa tärkeimmät ominaisuudet, kuten liikesarjojen generointi ja niiden tallen-

taminen, ja vasta sen jälkeen rakentaa käyttöliittymä sen ympärille, mitä sivustolla on mahdollista tehdä.

Ensimmäisessä käyttöliittymässä en panostanut kovin paljoa käytettävyyteen. Tärkeintä oli saada ensin tekninen toteutus valmiiksi, ja sen jälkeen keskittyä käyttöliittymän parantamiseen.

Toisella kierroksella paransin käyttöliittymää ja lisäsin ominaisuuksia sivustolle ja generaattoriin. Kolmannessa versiossa sain vasta käyttöliittymän sellaiseksi kuin halusin. Tässä vaiheessa oli jo tullut paljon tietoa siitä, miten generaattoria käytetään, ja mitä asioita pitää löytyä. Tästä oli mielestäni suuri hyöty. Käyttöliittymä sai enemmän hioutua itse oikeaan muotoonsa tarpeen mukaan kuin se, että olisin ensin suunnitellut kaiken ja sitten vasta toteuttanut. Samalla tuli tehtyä sivustolle uusia ominaisuuksia ja testattua niitä. Käyttöliittymä varmasti vielä muokkaantuu jatkossa paremmaksi, mutta nyt se on jo hyvällä mallilla.

### **2.3.1 Sivuston rakenne**

Pyrin noudattamaan mahdollisimman loogista ja yksinkertaista rakennetta. Samaan asiaan liittyvät sivut löytyisivät läheltä toisiaan ja navigaation yritän toteuttaa niin, että sivustolla pääsee liikkumaan mihin vain mahdollisimman vähillä klikkauksilla. Sivusto on jaoteltu karkeasti julkiseen ja yksityiseen puoleen.

Julkisella puolella on liikesarjoja arpova generaattori sekä jo olemassa olevien liikkeiden ja liikesarjojen haku. Yksityisellä puolella on mahdollista muokata mm. omia liikesarjoja sekä liikesarjalistoja.

Pääsivut ovat Etusivu, Generaattori, Kirjautuminen, Rekisteröinti, Palaute, Profiili, ja Haku. Haku-sivulla on hakuja eri asioihin sivustolla, kuten liikkeet, liikesarjat, liikelistat, liikesarjalistat. Näihin asioihin liittyvät toimintosivut, kuten esimerkiksi uuden liikkeen luominen, päivittäminen ja poistaminen löytyvät liikkeiden hakutulos-sivulta.

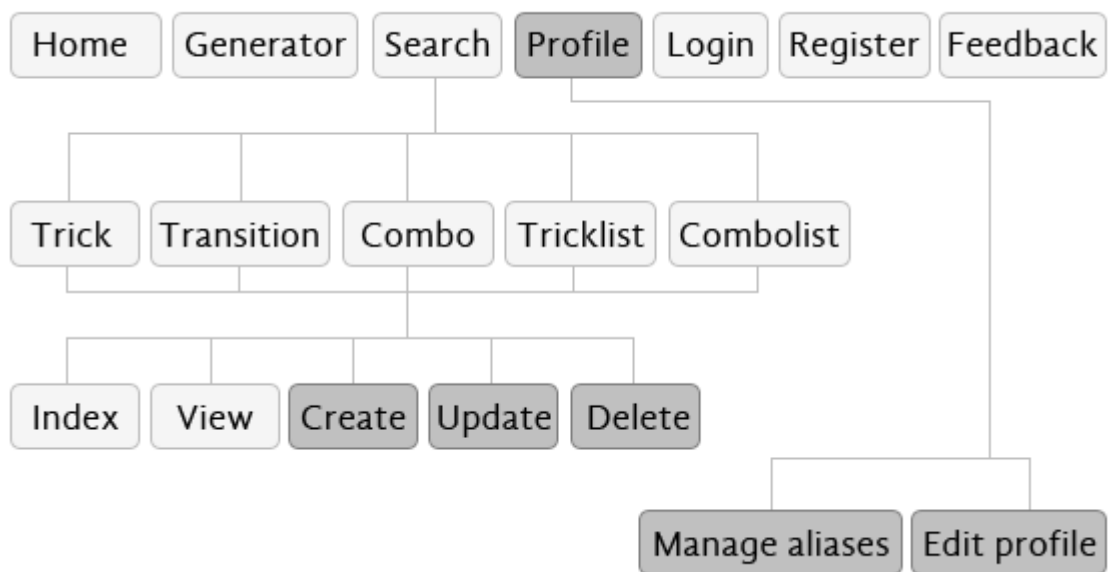


Profiilin alta löytyvät käyttäjän omat tiedot sekä niiden muokkaaminen. Profiiliin tulee pikalinkit käyttäjän itse lisäämiinsä listoihin ja liikkeisiin. Liikkeiden lyhenteiden eli aliasten muokkaaminen onnistuu myös profiilin kautta.

Generaattori-sivulla on liikesarjageneraattori ja samalta sivulta onnistuu myös generoitujen liikesarjojen muokkaaminen ja tallentaminen.

Palaute sivulle tulee lomake, jonka kautta voi lähettää palautetta combogeneraattorista.

Alustava sivustorakenne on esitelty kuvassa 2. Kuvassa vaalealla taustalla oleviin sivuihin on kaikilla käyttäjillä vapaa pääsy. Tummallalla taustalla oleviin sivuihin pitää olla rekisteröitynyt käyttäjä.



Kuva 2 Sivustorakenteen suunnitelma

### 3 KÄYTETYT TEKNIIKAT

Nyt kun käyttäjien tarpeet ovat selvillä ja on käsitys siitä, mitä sivustolla pitää olla, on aika valita tekniikat, joilla tämä kaikki on mahdollista toteuttaa. Käyttäjien toiveissa tuli monta kertaa esille älypuhelimet. Yksi mahdollinen vaihtoehto olisikin ollut toteuttaa combogeneraattori ohjelmana esimerkiksi androidille. Valitsin kuitenkin WWW-sivuston, koska halusin tavoittaa mahdollisimman monet käyttäjät ja WWW on tähän loistava ympäristö. Se toimii kaikilla laitteilla isoista pöytäkoneista pieniin älypuheliiniin. Haittapuolena on tietenkin se, että käytössä pitää olla internetyhteys. Eikä WWW-toteutus tule pärjäämään natiiviohjelmille nopeudessa tai käytettävyydessä. Kaiken lisäksi minun piti ottaa huomioon omat taitoni ja kiinnostukseni, jotka olivat vahvasti WWW-toteutuksen puolella. Seuraavaksi esittelen tekniikat, joita olen käyttänyt combogeneraattoria tehdessä.

#### 3.1 WWW-tekniikat

##### 3.1.1 HTML

HTML (*Hypertext Markup Language*) on merkkaukieli, jolla kuvataan WWW-sivujen rakennetta. HTML mahdollistaa mm. otsikoiden, tekstin, taulukoiden, listojen ja kuvien esittämisen WWW-sivulla. (W3C: HTML & CSS 2012)

##### HTML5

HTML-kielestä on parhaillaan tekeillä uusi versio HTML5. Uuden version on tarkoitus tulla W3C suositteluksi standardiksi vuoden 2014 viimeisellä neljänneksellä (W3C: Plan 2014. 2012)

HTML5 parantaa sivustojen semantiikkaa uusien elementtien myötä. Aikaisemmin sivujen eri osat merkittiin hyvinkin pitkälti div-elementtiä ja sopivaa luokkaa tai idtä käyttäen. HTML5 tarjoaa yleisimmille sivun osille omat elementit. Näitä ovat mm. article, section, nav ja footer. Parempi semantiikka yhdenmukaistaa sitä, kuinka selaimet lukevat ja näyttävät HTML-sivuja. (Dive into HTML5: What Does It All Mean? 2011)

Nimityksen HTML5 alle sisältyy paljon muutakin kuin uudet elementit. Uutta ovat esimerkiksi erilaiset ohjelmointirajapinnat (*Application programming interface, API*). Erilaiset rajapinnat ja elementit liittyvät toisiinsa. Esimerkkinä tästä on uusi videoelementti, jonka toisto, pysäytys ja muita painikkeita varten on tehty valmis JavaScript-rajapinta, josta näitä pääsee ohjelmoimaan. Tämä rajapinta antaa myös heti valmiiden toimintojen lisäksi paljon joustavuutta kehittäjälle. Video- ja audioelementtien lisäksi on paljon muitakin uusia ominaisuuksia, kuten canvas, offline storage, device access ja 3D-grafiikka. (Dive into HTML5: Detecting HTML5 features. 2011)

Vaikka HTML5 ei vielä olekaan virallinen standardi, voi uusia ominaisuuksia jo käyttää niiltä osin kun niitä tuetaan. HTML5-elementeistä ja ominaisuuksista on lista caniuse.com-sivulla. Sieltä voi tarkistaa, mitä uusia ominaisuuksia eri selaimet tukevat.

Toteuttamani sivuston kaikki sivut esitetään luonnollisesti HTML-kielellä. Käytin uusia HTML5-elementtejä sivuston rakenteen määrittelyssä sekä lomakkeissa. Muut HTML5 hienoudet saivat ikävä kyllä jäädä seuraavaan kertaan.

### 3.1.2 CSS

CSS (*Cascading Style Sheets*) on HTML- ja XML- (mukaan lukien erilaiset XML-kielet kuten XHTML ja SVG.) dokumenttien ulkonäön määrittelyyn tarkoitettu kieli. (Mozilla Developer Network: CSS 2012)

#### CSS3

Myös CSS:stä on parhaillaan tekeillä uusi versio CSS3. CSS3 tuo ja on jo tuonut paljon toivottuja ominaisuuksia WWW-kehittäjille. Uusien määrityksiensä avulla sivuista saadaan helpommin näyttävämpiä. Monet aikaisemmin JavaScriptiä tai purkkaviritelmiä vaatineet asiat on helppo toteuttaa. Tällaisia ominaisuuksia ovat mm. tekstin varjostus ja pyöreät kulmat. Kaikkia CSS3-määrittelyksiä ei vielä pääse käyttämään. Caniuse.com-sivustolta kannattaakin tarkistaa, mitkä selaimet tukevat mitäkin ominaisuuksia ja päätellä sen mukaan, kannattaako uutta ominaisuutta jo käyttää.

Sivuston elementtien asettelu ja värit ovat CSS-kielellä määriteltäviä. Käytin myös uudempiä CSS3-ominaisuuksia parantamaan sivuston ulkoasua, kuten nopeat häivytykset sivuston navigaatio elementtien taustaväreissä.

### 3.1.3 PHP

PHP (*PHP: Hypertext Preprocessor*) on skriptaus-kieli, jonka pääkäyttötarkoituksena on dynaamisten verkkosivujen ohjelmointi. PHP-syntaksi on ottanut vaikutteita C-, Java- ja Perl-kielistä. PHP:tä pidetään helposti opittavana kielenä. (PHP: Preface 2012)

PHP on erittäin yleinen, mm. Facebook, Wikipedia ja Flickr sekä monet suositut julkaisujärjestelmät kuten Drupal, Joomla! ja WordPress on tehty PHP:llä. (MacIntyre 2010. 1) Koulussa on opetettu dynaamisten verkkosivujen ohjelmointia PHP:ta käyttäen, joten tästä syystä omat taidot olivat selkeästi paremmat kaikkiin muihin kieliin nähden. Tästä syystä PHP oli helppo valinta palvelinpuolen kieleksi.

Combogeneraattorissa PHP:lla on toteutettu kaikki toiminnallisuus, mikä tapahtuu palvelimella. Mm. käyttäjienhallinta, tietokantahaut, generaattori sekä muut serveripuolen tehtävät ovat kaikki toteutettu PHP:tä käyttäen.

### 3.1.4 JavaScript

JavaScript on Netscapen vuonna 1995 kehittämä kieli WWW-sivujen käyttökokemuksen ehostamiseen. Aluksi JavaScriptiä käytettiin lähinnä efekteihin, kuten kuvan vaihtamiseen, kun kursori koskettaa sitä tai ponnahdusikkunoiden näyttämiseen. Nykyään JavaScriptillä voidaan tehdä paljon enemmän ja se on erottamaton osa WWW-sivuja. (Ullman 2012. 6-7)

JavaScriptin avulla saadaan toiminnallisuutta ja dynaamisuutta sivuille. Sivuja voidaan manipuloida monipuolisesti ilman, että niitä tarvitsee ladata uudelleen. Myös tiedon lähettäminen ja vastaanottaminen palvelimelle on mahdollista ilman sivun uudelleen lataamista Ajax-kutsujen kautta. Näin voidaan esimerkiksi rekisteröintilomakkeessa

tarkistaa, onko käyttäjänimi varattu, ja jos on, niin käyttäjä voi valita uuden nimen itselleen heti, eikä vasta lomakkeen lähetyksen jälkeen. (Ullman 2012. 9)

Combogeneraattorissa on käytetty paljon JavaScriptiä sivujen käyttökokemuksen parantamiseen. Suurin osa tästä toiminnallisuudesta on toteutettu jQuery-kirjastoa hyväksi käyttäen, josta kerron myöhemmin lisää.

### 3.1.5 MySQL

MySQL on erittäin suosittu, ja ilmainen avoimen lähdekoodin tietokannan hallintajärjestelmä. MySQL skaalautuu erinomaisesti, ja on nopea, ja kevyt käyttää. Tietokantaa voidaan siis kasvattaa sitä mukaan kun sivuston suosio ja tallennettavan tiedon määrä kasvaa. (MySQL 5.6 Manual: What is MySQL? 2013)

Suurin osa WWW-sivustoista käyttää tietokantaa. Tietokannat nimensä mukaisesti sisältävät tietoa. Esimerkiksi keskustelufoorumilla kaikki viestit ja viestiketjut ovat tallennettuna tietokantaan, ja haetaan sieltä käyttäjän nähtäville tarpeen mukaan.

Tietokanta on erittäin tärkeä osa combogeneraattoria. Lähes kaikki sisältö ja kaikki muunneltavissa oleva tieto on tallennettuna tietokantaan. Tietokannasta löytyvät mm. liikkeet, liikesarjat, liikelistat, liikesarjalistat sekä käyttäjien tiedot.

### 3.2 Ohjelmistokehykset

Ohjelmistokehys (*Software Framework*) on paketti valmista koodia, joka ratkaisee yleisimpiä ongelmia ja näin nopeuttaa työskentelyä. Ohjelmistokehyksiä on joka lähtöön ja jokaiselle kielelle. Pienimmillään ohjelmistokehys voi olla hyvä pohja uudelle ohjelmalle tai WWW-sivulle. Suurimmillaan ohjelmistokehyksen komponentteja yhdistelemällä saadaan aikaan kaikki tarvittava. Seuraavaksi käyn läpi, mitä ohjelmistokehyksiä käytin apuna sivuston toteuttamisessa.

### 3.2.1 Yii

Yii on PHP:llä kirjoitettu ohjelmistokehys, jolla on mahdollista rakentaa nopeasti suuria ja skaalautuvia WWW-sivustoja. Yiiin perustaja Qiang Xue aloitti Yiiin kehittämisen vuonna 2008. (Winesett 2010, 7)

Erilaisia php-ohjelmistokehyksiä löytyy kymmeniä, ja niille kaikille löytyy oma käyttäjäkuntansa. Suurimmat ohjelmistokehukset ovat kaikki niin kehittyneitä, että ne varmasti tyydyttävät vaativampaakin käyttäjää. On siis omien mieltymysten mukaista valita itselleen sopiva ohjelmistokehys.

Minulla ei ollut aikaa käydä kaikkia ohjelmistokehyksiä läpi ja tehdä kaiken kattavia vertailuja. Enhän edes tarkkaan tiennyt, mitä tarvitsen. Luin ammattilaisten kirjoittamia blogeja saadakseni yleiskäsityksen tarjonnasta. Vertailuja tehdessäni löysin monta hyvää ehdokasta. Keskusteluista nousi pinnalle se, että ei ole niin kovin suurta väliä, minä ohjelmistokehysten valitsee, kunhan siitä löytyy seuraavat ominaisuudet: ohjelmistokehysten olisi hyvä noudattaa MVC-arkkitehtuuria, koodin pitäisi olla PHP5-yhteensopivaa, ohjelmistokehuksesta pitäisi olla saatavilla kattava ja selkeä dokumentaatio, ja ongelmatilanteissa aktiivinen keskustelupalsta voi toimia pelastajana.

Yii onnistui herättämään kiinnostukseni sekä uteliaisuuteni, ja kun tekniset asiat eivät estäneet minua valitsemasta Yiitä, päädyin siihen. Osittain siis Yiiin imago ja markkinapuheet vaikuttivat päätökseeni.

Haluan oppia kirjoittamaan hyvää PHP-koodia ja Yiitä käyttämällä näen, miten asiat hoidetaan ammattilaisten tyyliin. Yiissä on otettu huomioon myös turvallisuus. Jos olisin itse tehnyt kaiken alusta asti, olisi sivustoon varmasti jäänyt tietoturva-aukkoja.

Combogeneraattori on rakennettu täysin Yiiin päälle. Työtä tehdessä pyrin toteuttamaan sivuston mahdollisimman paljon ns. Yii-tyylillä. Eli käyttää kaikkea valmista, mitä Yii tarjoaa. Yiiin eri ominaisuuksien opetteluun menikin paljon aikaa.

### 3.2.2 jQuery

jQuery on erittäin suosittu JavaScript-kirjasto. Se nopeuttaa ja helpottaa JavaScript koodin kirjoittamista. jQuerystä löytyy toiminnot lähes kaikkeen, mitä normaalisti WWW-sivuilla tarvitaan. HTML-dokumentin elementtien muokkaaminen, tapahtumien käsittely, animaatiot ja Ajax-kutsut sekä paljon muuta onnistuu jQueryn valmiilla funktioilla. (jQuery: What is jQuery? 2013.)

Yiissä on toteutettuna Zii-niminen widget-kirjasto, joka sisältää suuren osan jQuery ui-toiminnoista. Nämä widgetit helpottavat jQuery uin ominaisuuksien käyttöä Yiissä muuttamalla ne samaan muotoon kuin muutkin Yiin widgetit. (Yii Framework: Class Reference. 2013.)

Combogeneraattorissa käytin paljon jQueryä. jQueryn Ajax-toiminto on ahkerassa käytössä mm. generaattorin liikkeiden generoinnissa, muokkaamisessa sekä tallentamisessa.

### 3.2.3 Bootstrap

Bootstrap on ohjelmistokehys asiakaspuolen kehittämiseen. Bootstrapista löytyy valmiina erinomainen ratkaisu sivuston ulkoasun rakenteelle sekä paljon pieniä hyödyllisiä ominaisuuksia, jotka helpottavat käyttöliittymän rakentamista. Bootstrapissa on myös hyvän näköinen oletusulkoasu, ja se on hyvä lähtökohta oman ulkoasun toteuttamiseen. (Bootstrap: Documentation. 2013.)

Käytin bootstrapia sivuston rakenteen määrittelyyn. Samalla sain ulkoasusta valmiiksi responsiivisen. Käytin myös Bootstrapin valmiita elementtejä ja määrittelin niille tarvittaessa uudet sivuston muuhun ulkoasuun sopivat tyylit. Ulkoasun rakentaminen sujui huomattavasti nopeammin Bootstrapin avulla.

### 3.2.4 LESS

LESS on laajennus CSS-kielelle. LESS tuo CSS-kieleen ohjelmoinnista tuttuja ominaisuuksia, kuten muuttuja ja funktiot. Lisäksi LESSiä käyttäen on mahdollista luoda sekoituksia, joilla voidaan määrittää vaikkapa kaikki selainten etuliitteet (*vendor prefix*) samaan sekoitukseen. Tämä lyhentää ja selkeyttää CSS-koodia. Kun on tarve muuttaa jotain tietoa, käy se kätevästi yhdestä paikasta. (LESS: Documentation. 2013.)

Bootstrapissä on käytetty LESSiä tyylien määrittelyyn. Pystyn itse hyödyntämään Bootstrapissa valmiiksi määriteltyjä sekoituksia, muuttujia ja funktiota. Nämä nopeuttivat ulkoasun tekemistä ja samalla sain paremmin eri selaimilla toimivia määrittelyitä. LESSistä on siis paljon hyötyä, kun käytössä on Bootstrap ohjelmistokehys.

LESS-koodi käännetään erikseen kääntäjällä selainten ymmärtämäksi CSS-kieleksi. Combogeneraattorissa kaikki CSS-määrittelyt on kirjoitettu ensin LESS-muotoon ja käännetty sitten CSS-kielelle.



## 4 SIVUSTON TOTEUTTAMINEN

Tässä luvussa käyn läpi, miten toteutin sivuston. Nyt kun vaatimukset jo on määriteltyinä ja tekniikat valittuna, on aika käydä kunnolla itse toteutuksen kimppuun. Jaoin sivuston rakentamisen kahteen osaan; yleiset asiat, jotka sopivat mihin tahansa muuhunkin sivustoon sekä trikkaukseen liittyvät asiat, jossa käsittelen hieman trikkauksen teoriaa ja tälle sivustolle ominaisia asioita. Pysin käymään toteutuksen vaiheet siinä järjestyksessä, kun niitä tein, vaikkakin toteutuksen aikana eri vaiheet sulautuivat toisiinsa paljon.

Sivuston perustana on PHP-ohjelmistokehys Yii. Käyn ensin Yiin toiminnan läpi ja sitten järjestyksessä sivuston rakentamisen pala palalta aina valmiiksi sivustoksi asti.

### 4.1 Yii-projektin perusrakenne ja toiminta

Yii käyttää MVC-arkkitehtuuria, joka on hyvin suosittu WWW-ohjelmoinnissa. MVC on lyhenne sanoista Model (*malli*) View (*näkymä*) ja Controller (*ohjain, kontrolleri, käsittelijä*). MVC-arkkitehtuurin tarkoituksena on pitää koodi selkeässä järjestyksessä ja jokainen osio mahdollisimman itsenäisenä. (Xue 2012, 19)

#### **Malli**

Malli edustaa dataa. Mallin koodi on vastuussa tiedon käsittelystä. Sen kautta tapahtuu mm. tietokantaan tallentaminen sekä tietokannasta lataaminen.

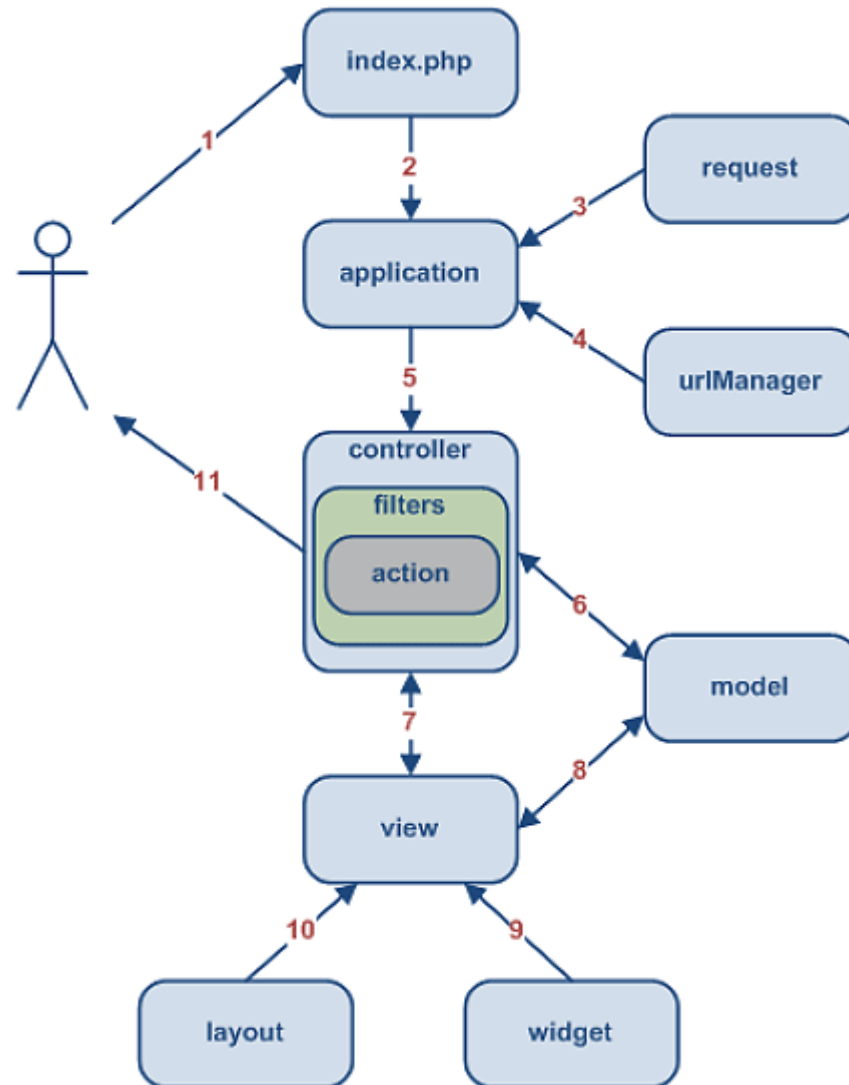
#### **Näkymä**

Näkymä on tarkoitettu täysin tiedon näyttämiseen. Näkymän pitäisi sisältää vain käyttöliittymään liittyvää koodia.

#### **Ohjain**

Ohjain on ohjelman aivot. Se toimii mallin ja näkymän yhdistävänä tekijänä. Se käsittelee käyttäjän pyynnöt ja päättää, mitä tarvitaan tietokannasta, ja minkälainen näkymä valitaan.

MVC-arkkitehtuurin lisäksi Yiiissä on vielä etu-ohjain, jota kutsutaan applikaatioksi. Applikaatio ottaa vastaan käyttäjän lähettämän pyynnön ja ohjaa sen oikealle ohjaimelle. (Xue 2012, 19) Yii applikaation tyypillinen toiminta on kuvattu kuvassa 3.



Kuva 3 Yii applikaation tyypillinen toiminta (Xue 2012, 20)

1. Käyttäjä menee selaimellaan osoitteeseen `http://www.example.com/index.php?r=post/show&id=1` ja palvelin käsittelee pyynnön ajamalla esilatausohjelman (*bootstrap script*) `index.php`-tiedostossa.
2. Esilatausohjelma luo applikaation ja ajaa sen.
3. Applikaatio saa yksityiskohtaiset tiedot käyttäjän pyynnöstä `request`-nimiseltä applikaatio-komponentilta.
4. Applikaatio ottaa selville pyydetyn ohjaimen (*controller*) ja toiminnon (*action*) käyttäen apunaan `urlManager`-applikaatio komponenttia. Esimerkissä ohjain

on `post`, joka viittaa `PostController`-luokkaan. Toiminto on `show`, jonka tarkoitus riippuu ohjaimesta.

5. Applikaatio luo ilmentymän pyydetyistä ohjaimista, jotta se voi käsitellä käyttäjän pyyntöä eteenpäin. Ohjain määrittää, että toiminto `show` viittaa ohjaimen luokassa olevaan metodiin nimeltä `actionShow`. Tämän jälkeen luodaan ja ajetaan toimintoon liittyvät suodattimet, ja jos suodattimet sen sallivat, toimintomethodi ajetaan.
6. Toiminto metodi lukee `Post`-mallin tietokannasta, jonka `id` on 1.
7. Toiminto muotoilee näkymän nimeltä `show Post`-mallin kanssa.
8. Näkymä lukee ja näyttää `Post`-mallin attribuutit.
9. Näkymä käynnistää tarvittavat widgetit.
10. Näkymä muotoilee tuloksen, joka on upotettuna ulkoasuun.
11. Toiminto saa näkymän muotoilun valmiiksi ja näkymä näytetään käyttäjälle.

## 4.2 Alkutoimenpiteet

### Yiin asennus

Uusin versio ladataan Yiin sivuilta ja tiedostot asetetaan palvelimelle kansioon, johon on pääsy WWW:n kautta. Asentamisen jälkeen voi tarkistaa Yiin yhteensopivuuden palvelimen kanssa osoitteesta <http://hostname/yii/requirements/index.php>

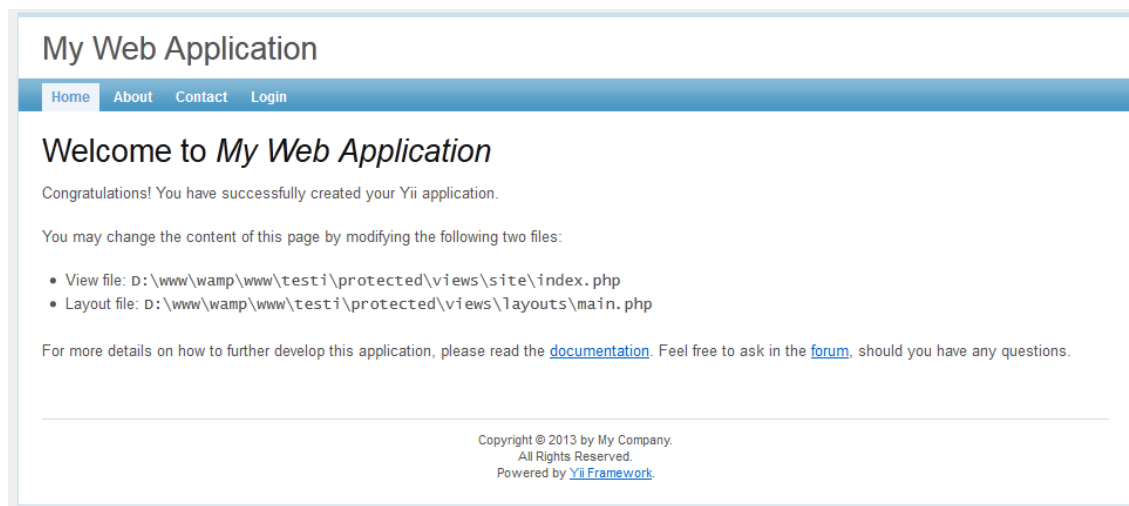
Yiin ei kuitenkaan ole pakko olla kansiossa, johon on pääsy webin kautta. Yii toimii `index.php`-tiedostonsa kautta, jossa esilatausohjelma sijaitsee, ja se on ainoa tiedosto, jonka on pakko olla WWW-käyttäjien saatavilla. Tämä antaa paremman suojan hakkeireita vastaan. (Xue 2012, 5)

### Uusi projekti

Uuden Yii-projektin luonti onnistuu parhaiten komentoriviltä, käyttäen Yiin komentorivityökalua nimeltä `yiic`. `Yiic` on pätevä työkalu monenlaiseen työhön Yiitä käyttäessä, mutta minä käytin sitä vain uuden Yii-projektin luomiseen.

Komento `yiic webapp polku/haluttuun/kansioon/ProjektinNimi` luo kattavan kansiorakenteen ja kaikki tarvittavat tiedostot. Uutta projektia pääsee katsomaan

menemällä selaimella kansioon, johon uusi Yii-projekti luotiin. Sivulla on joitakin ominaisuuksia valmiina, kuten yhteydenottolomake ja mahdollisuus sisään kirjautumiseen. Kuvassa 4 näkyy oletusprojektin etusivu.



Kuva 4 Yii-projekti selaimesta nähtynä (Xue 2012, 8)

## Tietokanta

Combogeneraattoriin liittyy paljon erilaista tietoa ja paras tapa tiedon säilyttämiseen on tietokanta. Liikkeet, liikesarjat, liikelistat, liikesarjalistat, käyttäjien tiedot ja monta muuta tärkeää asiaa tallennetaan tietokantaan.

Tietokannan tein phpMyAdmin-ohjelmalla. Tietokannassa pitää olla myös käyttäjä, jonka luominen sujuu myös phpMyAdmin-ohjelmaa käyttäen. Yiissä tietokantayhteyden määrittäminen tapahtuu `protected/config/main.php`-tiedostosta applikaatio-komponenttia `db` muuttamalla. Yii tarvitsee tietokannan osoitteen, käyttäjänimen sekä salasanan. Myös muita tietoja voi antaa, ja yhteyden ottaminen useampiin tietokantoihin on mahdollista. Koodiesimerkissä 1 on tarvittava koodi tietokantayhteyden muodostamiseen MySQL-tietokantaan.

```
'components'=>array(
    .....
    'db'=>array(
        'connectionString' => 'mysql:host=localhost;dbname=tietokanta',
        'username' => 'nimi',
        'password' => 'salasana',
        'charset' => 'utf8',
    ),
    .....
),
```

Koodiesimerkki 1. Tietokantayhteyden määrittäminen applikaatiokomponentilla (Xue 2012, 79)

### Tarvittavien laajennusten asentaminen

Laajennukset (*extensions*) ovat koodia, joka tuo lisätoiminnallisuutta Yii-ohjelmiin. Erilaisia laajennuksia on jo toista tuhatta ja ne ovat listattuina Yiiin sivuilla. (Yii framework: Extensions. 2013.) Hyvin toteutettu laajennus voi säästää paljon vaivaa ja aikaa.

Usein samasta aiheesta tulee vastaan useampia hyvin samankaltaisia laajennuksia. Osa on paremmin toteutettuja kuin toiset. Onkin siis tärkeää valita niistä omaan projektiin parhaiten sopiva. Yiiin sivuilla on pisteytysjärjestelmä, jossa voi antaa, joko plus tai miinus pisteen laajennukselle. (Yii framework: Extensions. 2013.) Siitä näkee nopeasti, mitä meiltä yhteisö on laajennuksesta. Laajennusta valittaessa olisi tärkeää ottaa huomioon, kuinka aktiivisesti laajennusta kehitetään, ja onko löydettyjä ohjelmointivirheitä korjattu. Yleensä laajennukselle on viestiketju keskustelupalstalla, josta saa lisää tietoa ja apua ongelmatilanteissa.

Asensin seuraavat laajennukset:

**Giix:** paranneltu versio Yiiin koodigeneraattorista Giistä.

**Yii-user:** toteuttaa käyttäjien hallintaan tarvittavat asiat.

**Auth:** käyttäjien oikeuksien hallintaan.

**Phpass:** tehokas salasanan salaaja.

**Yii-Booster:** iso kirjasto valmiita käyttöliittymä komponentteja.

Kerron kustakin laajennuksesta myöhemmin lisää, kun ko. laajennusta käytetään.

Laajennusten asentamisessa on Xuen (2012, 150) mukaan seuraavat kolme vaihetta:

1. Koodin lataaminen Yiin sivuilta. [www.yiiframework.com/extensions](http://www.yiiframework.com/extensions)
2. Koodin sijoittaminen extensions-kansioon laajennuksen omalla nimellä tehtyyn alikansioon.
3. Koodin tuonti Yiihin. Laajennuksen konfigurointi ja lopuksi laajennuksen käyttäminen.

Näistä viimeinen kohta kaivannee hieman selittämistä. Koodin tuonti Yiihin tarkoittaa `config/main.php`-tiedostossa `import`-kohdan muuttamista niin, että Yiiille kerrotaan, mitä koodia halutaan ottaa mukaan ohjelman suoritukseen. Konfigurointi on usein sitä, että lisätään applikaatiokomponentteihin laajennus `main.php`-tiedostossa ja annetaan siellä asetuksia, jotka määrittävät laajennuksen toimintaa. Esimerkiksi `yii-user`-laajennuksessa voidaan mm. määrittää, lähetetäänkö käyttäjälle aktivointisähköposti rekisteröitymisen jälkeen. Laajennuksen käyttäminen vaatii myös omanlaisensa koodin, joka tulee siihen ohjelman osaan, jossa sitä halutaan käyttää.

### 4.3 Trikkaukseen liittyvät asiat

Tässä alaluvussa käsitelen aluksi hieman trikkauksen teoriaa. Käyn läpi perusliikkeen eri vaiheet. Liikkeen vaiheiden ymmärtäminen on erittäin tärkeää, jotta valmis generaattori tuottaa loogisia liikesarjoja. Kerron, mikä idea on liike- ja liikesarjalistoissa. Tämän jälkeen kerron, kuinka toteutin tietokannan, joka mahdollistaa näiden asioiden tallentamisen ja käyttämisen. Tietokannan tietojen käsittelyyn tarvitaan oma ohjelmakoodinsa, jonka tekemisen hoidamme Yiin koodigeneraattorilla. Lopuksi näitä tietoja hyväksikäyttäen selitän, kuinka liikesarjojen generointi tapahtuu, ja kuinka käyttäjä voi muokata ja tallentaa juuri generoituja liikkeitä.

#### 4.3.1 Liikkeet ja liikesarjat

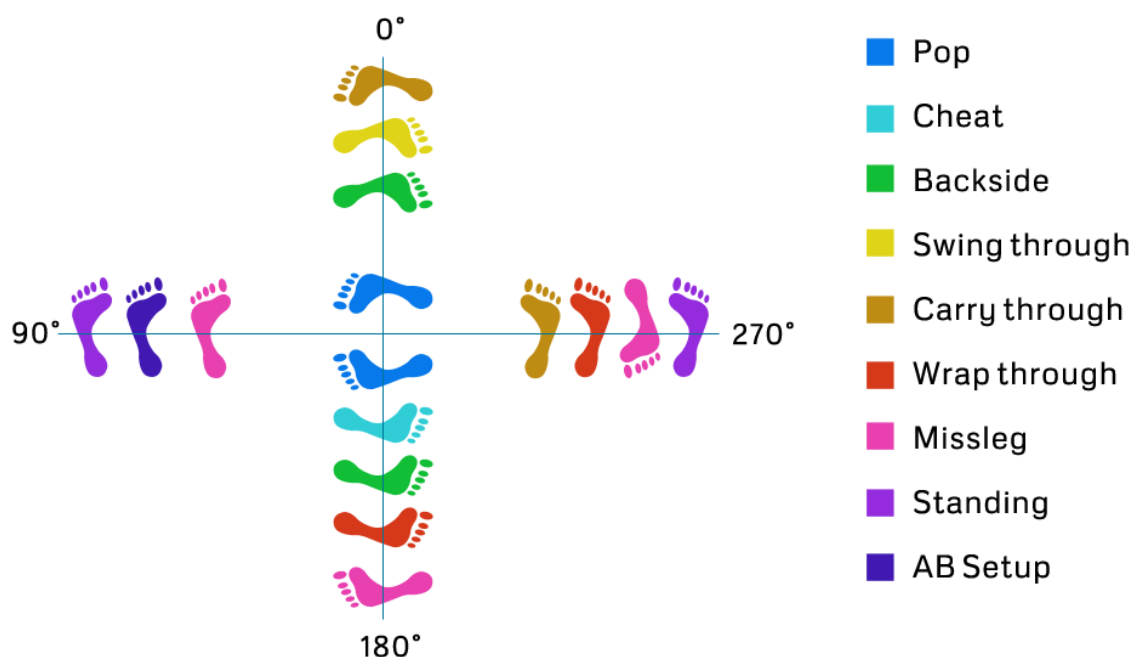
Trikkauksessa on kymmeniä eri perusliikkeitä sekä niistä johdettuja useita variaatioita. Liikkeet jaetaan karkeasti ottaen kolmeen eri luokkaan potkut, kierteet ja voltit. Lisäsin

myös Combogeneraattoriin setup-tyyppiset liikkeet. Setup-liikkeistä saadaan lisää vauhtia ja voimaa seuraavaan liikkeeseen. Niitä harvoin tehdään ilman toista liikettä.

Generaattorin vuoksi on tärkeää tietää, mitä eri vaiheita liikkeissä on. Olen jakanut liikkeiden suorittamisen kolmeen vaiheeseen. Aloitusasento, liikkeen suorittaminen, sekä lopetusasento. Tämä sen vuoksi, että saan generaattorin toimimaan helposti katsomalla liikkeiden eri aloitus- ja lopetusasentoja. Kaksi liikettä voidaan yhdistää, jos ensimmäisen liikkeen lopetusasento on sama kuin seuraavan liikkeen aloitusasento. Tätä liikkeiden välissä olevaa aloitus- ja lopetusasennon yhdistelmää kutsutaan yleisesti nimellä transiio, jolla kuvataan sitä, kuinka liikkeestä siirrytään toiseen liikkeeseen liikesarjan suorittamisen aikana.

Combogeneraattorissa aloitusasunnoista löytyvät täsmälleen samat tiedot kuin lopetusasunnoistakin. Aloitusasento on siis transiio ennen liikettä ja lopetusasento on transiio liikkeen jälkeen.

Transiioita on useita erilaisia. Olen valinnut generaattoriin yhdeksän yleisintä, joilla saadaan määriteltyä useimmat yleisesti tehdyt liikesarjat. Tein eri asunnoista havainnollistavan kuvan (kuva 5), kun suunnittelin liikkeiden aloitus- ja lopetusasentojen määrittämistä.



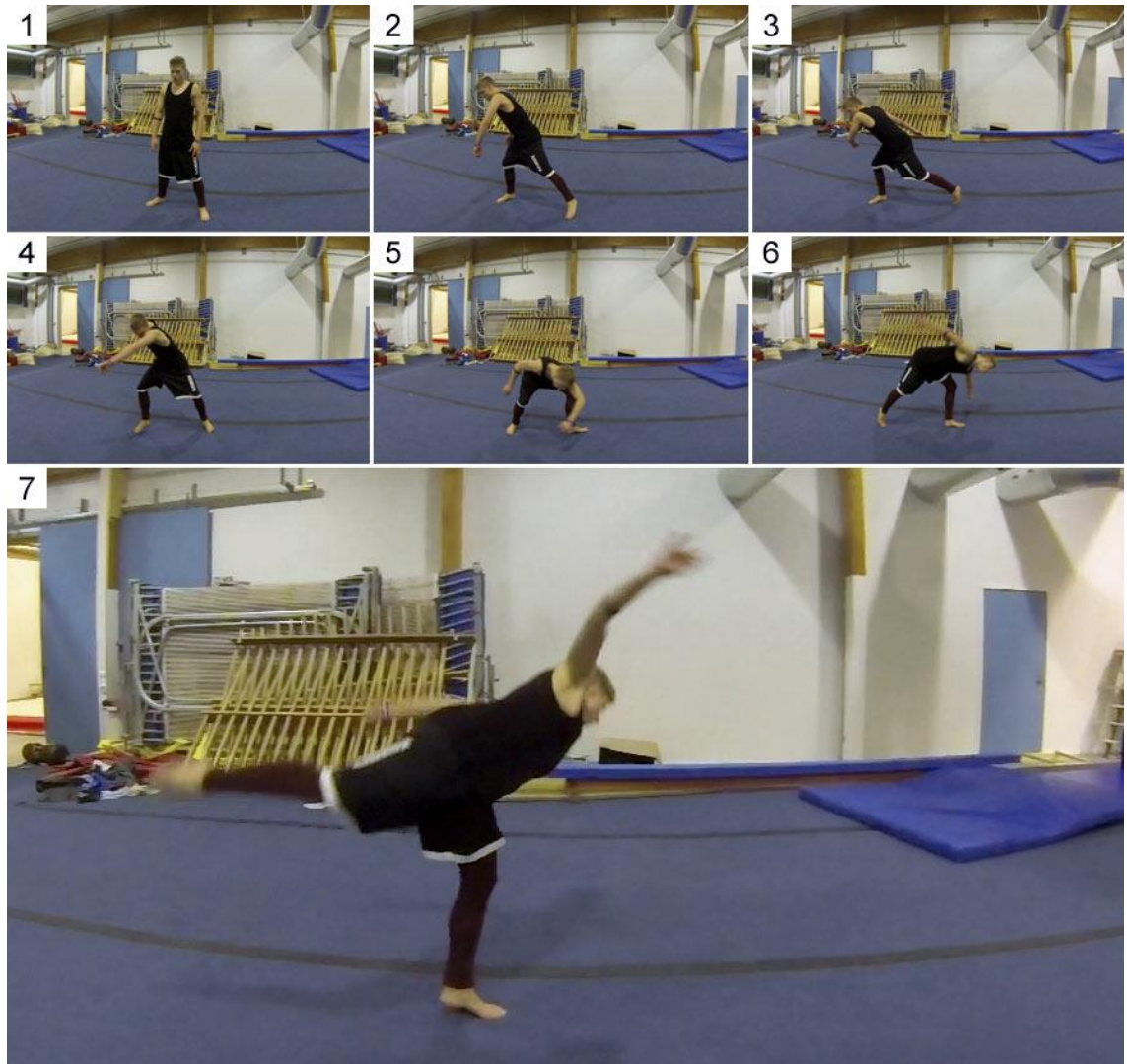
Kuva 5 Yleisimmät aloitus- ja lopetusasennot eli transiitot

Värit määrittävät, mitkä jalat kuuluvat mihinkin asentoon. Kuvaan on merkitty aina maata koskettava jalka. Jos sama jalka näkyy useamman kerran se tarkoittaa, että jalka voi olla vain yhdessä näistä kohdista asennon aikana. Jos näkyy sekä vasen, että oikea jalka, täytyy molempien olla maassa samaan aikaan. Eli liikkeen suorittaminen lähtee tasajalkaa. Kuvassa 5 olevan jalan asento näyttää, mihin suuntaan triikkaajan tulisi olla. Jalka ei siis tarkoita tässä kuvassa sitä, mihin suuntaan varpaat osoittavat. Jalan asento riippuu liikkeen suorittajasta.

Hyvin harvoin triikkaajan jalat ovat tarkalleen oikeaan suuntaan. On myös mahdollista, että liikesarja ei mene suoraan, vaan pahimmassa tapauksessa kiertää ympyrää. Generaattoria varten kaikkien liikesarjojen oletetaan menevän samaan suuntaan ja aina suoraan. Generaattori sallii myös väliaskeleet liikkeiden ja transitioiden välissä. Tämä mahdollistaa useampien liikkeiden yhdistämisen, kun triikkaaja voi liikkeen suorittamisen jälkeen ottaa väliaskeleen ennen kuin seuraavan liikkeen suorittaminen alkaa. Kaikissa transiatioissa ei kuitenkaan tarvitse ottaa ylimääräisiä väliaskeleita ja päätös väliaskeleiden tarpeellisuudesta jääkin liikesarjan tekijälle.

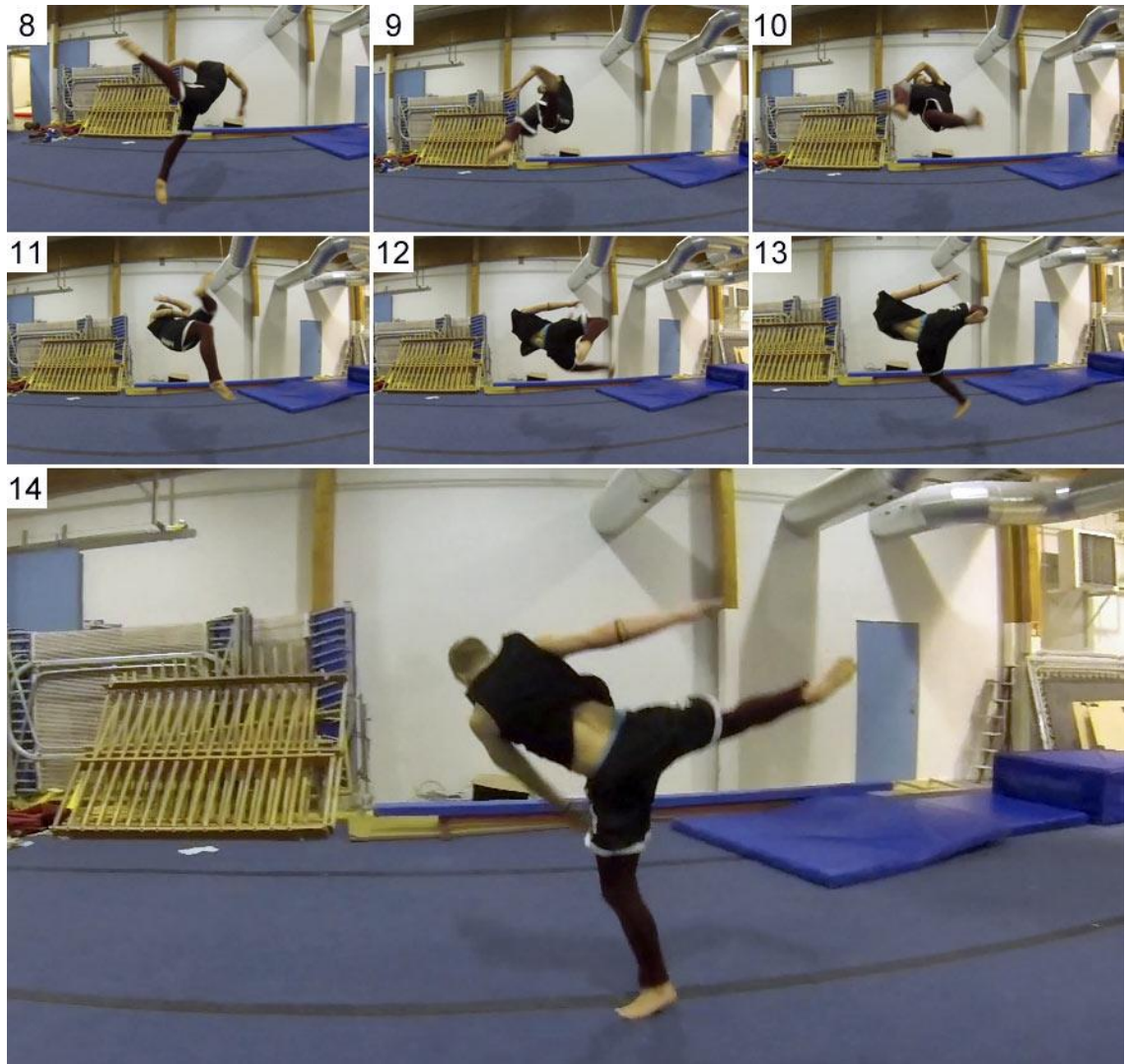
Kuinka tämä sitten toimii käytännössä. Kuvassa 6 on esimerkki perusliikkeestä nimeltä butterfly twist.





Kuva 6 Ensimmäinen osa liikkeestä b-twist

Ensimmäisessä pikkukuvassa näkyy normaali seisominen. Pikkukuvissa 2–6 näkyy b-twistille tavanomaista vauhdinottoa. Kohdassa 7 näkyy absetup-lähtöasento. Kirjaimet a ja b tarkoittavat aerial- ja b-twist-liikkeiden lähtöasentoa. Vasemmalle puolelle liikettä tehtäessä on triikkaajan vasen jalka maassa ja oikea jalka ilmassa. Vaikka alussa näyttääkin siltä, että molemmat jalat ovat maassa. Aloitusasento katsotaan siitä kohtaa, missä triikkaaja on lähdessä suorittamaan liikettä. Kuvassa 7 jatkamme liikkeen analysointia.



Kuva 7 Toinen osa liikkeestä b-twist

Pikkukuvat 8–13 ovat liikkeen suorittamista. Kohdassa 14 näkyy ensimmäinen mahdollinen lopetusasento. Vasen jalka koskettaa maata ja liikkeen suorittaminen katsotaan loppuneeksi. Ensimmäinen mahdollinen transitio olisi swing through. Swingissä oikea jalka heilahtaa vasemman vierestä nostaen triikkaajan takaisin ilmaan. Tämän sijasta voidaan kuitenkin siirtyä luonnollisesti useampiin lopetusasentoihin, joista voidaan tehdä uusi liike. B-twistin jälkeen voidaan tehdä liikkeitä, joiden aloitusasento on jokin seuraavista: standing, backside, absetup, cheat, missleg tai swing through. Tavallisesti liikkeissä on enemmän lopetusasentoja kuin aloitusasentoja. Kuvassa 8 näkyy loput liikkeen kulusta.



Kuva 8 B-twist liikkeen loppu

Lopuksi näemme vielä kuvista luonnollisen liikeradan takaisin siihen, mistä lähdettiin. Uusi aloitusasento eli transitio voi alkaa nyt missä tahansa vaiheessa se vain on luonnollista. Generaattori sallii myös väliaskeleen ottamisen liikkeiden aloitus- ja lopetusasentojen välillä.

### Liikkeiden nimet

Vaikka liikkeille löytyykin lähes standardin mukaiset nimet, löytyy erilaisia nimeämistyylejä paljon. Halusin tarjota mahdollisuuden käyttää omia nimiä eri liikkeille. esimerkiksi liike butterfly twist kääntyy monen triikkaajan suussa b-twistiksi. Liike cheat 900 kick on monen mielestä helpommin ilmaistuna c9. Näissäkin on vielä eroja, kun ne kirjoitetaan. Joku kirjoittaa vain c9 ja toinen pitää enemmän nimestä cheat 9 ja joku nimeää c900. Jotta generaattorin käyttö tuntuisi mahdollisimman tutulta, päätin tarjota jokaiselle mahdollisuuden nimetä liikkeet niin kuin he haluavat. Myös samoille liikkeille on erilaisia nimiä. Esimerkiksi 540 twist ja cheat 720 twist ovat molemmat täysin sama liike, mutta vain eri nimellä.

Usein liikkeen nimi vaihtuu, jos aloitusasentoa vaihdetaan. Esimerkki 540 kick lähtee cheat-asennosta, mutta jos se tehdään pop-asennosta, vaihtuu sen nimeksi pop 540 kick. On liikekohtaista, lisätäänkö aloitusasennon nimi liikkeen eteen. Myös jotkin aloitusasennot ovat sellaisia, että niitä ei koskaan mainita liikkeiden nimistä puhuttaessa.

### 4.3.2 Liikelistat

Liikelistojen tarkoituksena on välittää generaattorille vain ne liikkeet ja transitiot, joita käyttäjä haluaa generoidussa liikesarjassaan nähdä. Liikelista koostuu liikkeistä ja transiioista. Generaattorin käyttäjien taso vaihtelee aloittelijasta ammattilaiseen, joten on tarpeen valita generaattorin arvottavaksi vain niitä liikkeitä, joita käyttäjä itse osaa, tai

haluaa tehdä. Toisinaan halutaan tehdä liikesarjoja, joissa on esimerkiksi pelkkiä voltteja. Tällöin generaattoriin voidaan antaa liikelista, jossa on pelkkiä voltteja.

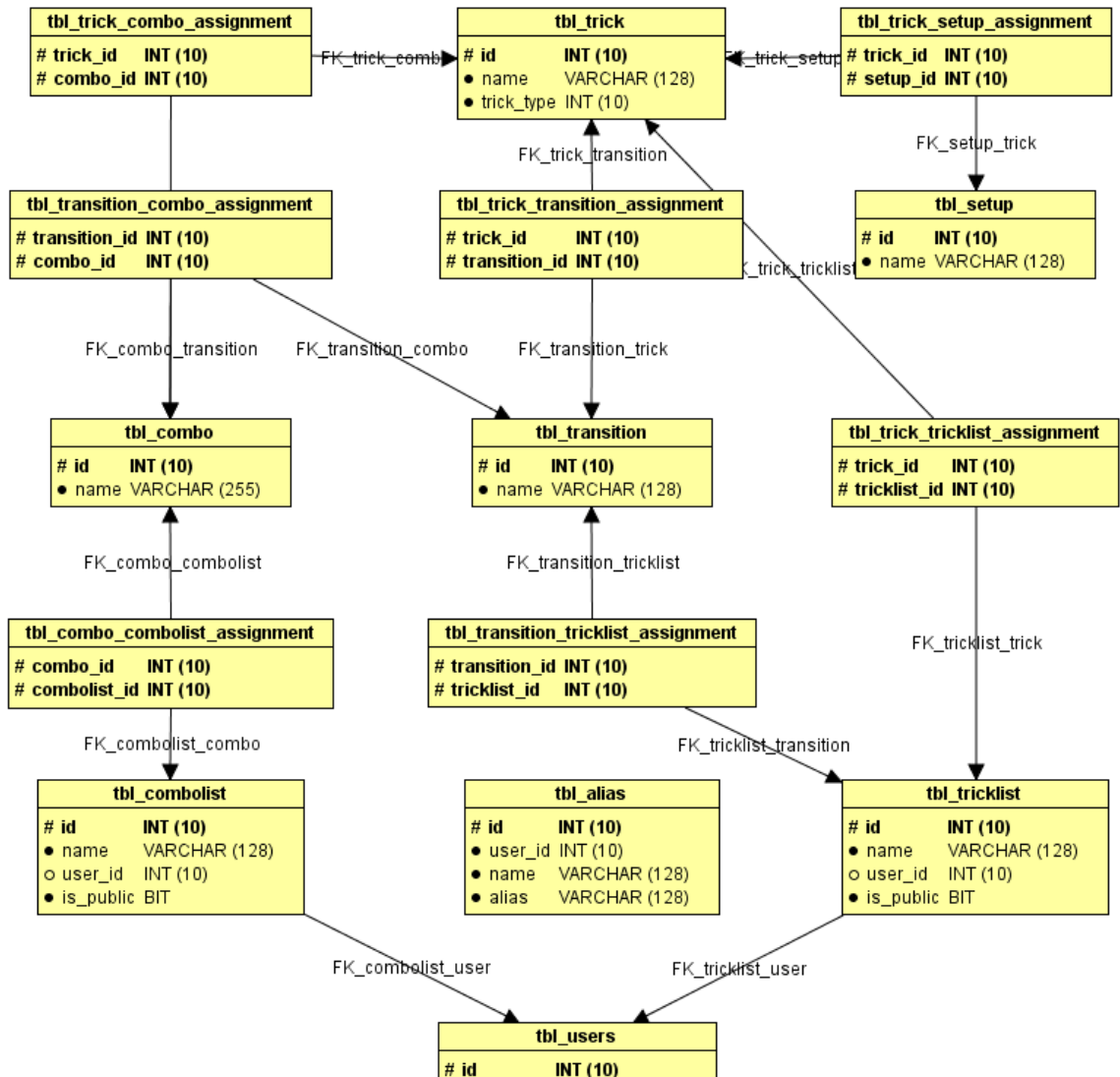
Liikelistoja voi tehdä itselleen niin monta kuin näkee tarpeelliseksi. Listat voi määrittää julkisiksi tai yksityisiksi. Julkisia listoja voivat muutkin sivuston käyttäjät käyttää generaattorissa, mutta listan muokkaaminen on vain listan omistajalle.

### **4.3.3 Liikesarjalistat**

Liikesarjalistoihin voi tallentaa generaattorin arpomia liikesarjoja myöhempää käyttöä varten. Liikesarjalistoihin käyttäjä voi lisätä myös muualta sivustolta löytämiään liikesarjoja. Käyttäjällä voi olla useita liikesarjalistoja ja niitä voi asettaa julkisiksi tai yksityisiksi.

### **4.3.4 Tietokanta**

Tietokanta toimii tallennus- ja säilytyspaikkana kaikille äsken mainituille asioille. Jokainen näistä tarvitsee oman taulunsa tietokantaan. Seuraavaksi käyn läpi, mitä tauluja tietokantaan piti tehdä, ja kuinka taulujen väliset suhteet on toteutettu. Kuvassa 9 on näkymä tietokannan rakenteesta.



Kuva 9 Tietokannan rakenne

Perustoiminnallisuutta varten tarvitaan seuraavat taulut: **tbl\_trick** (liike), **tbl\_setup** (aloitusasennot), **tbl\_transition** (lopetusasennot), **tbl\_combo** (liikesarjat), **tbl\_tricklist** (liikelistat), **tbl\_combolist** (liikesarjalistat). Tietokannassa on valmiiksi taulut käyttäjien tiedoille, jotka yii-user-laajennus loi asennuksen aikana. Näistä näkyy kuvassa 9 vain **tbl\_users** ja siitäkin vain id.

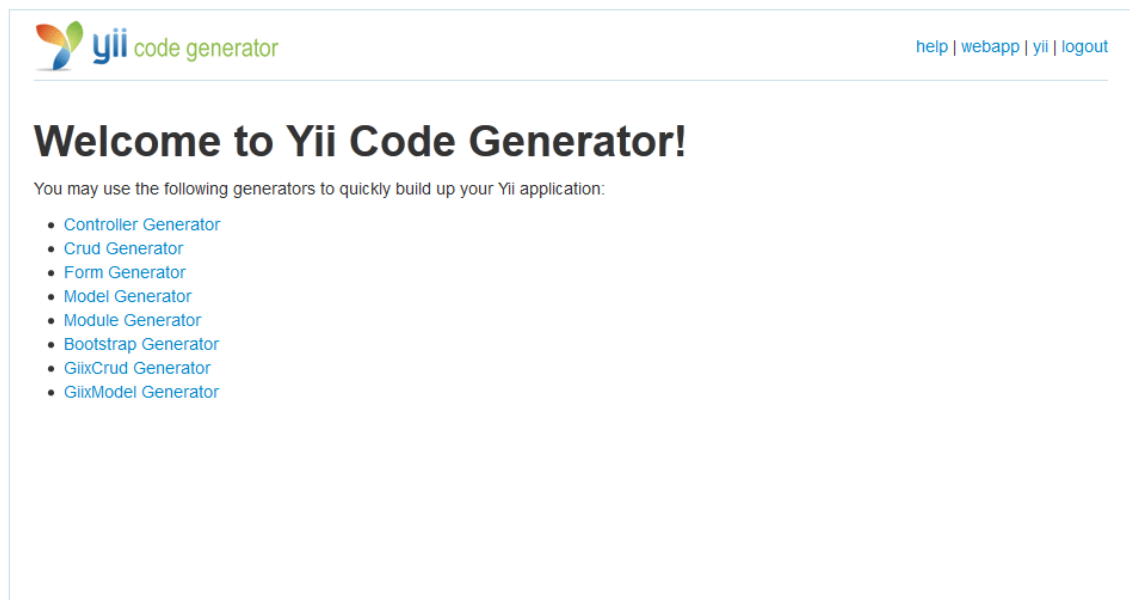
Tietokannassa monien taulujen tiedot liittyvät oleellisesti toisiinsa. Esimerkiksi liike tarvitsee aloitus- ja lopetusasentoja. Liikkeellä ja asennoilla on niin sanottu monen suhde moneen-relaatio (*many-to-many*). Yhdellä liikkeellä voi olla useampia aloitusasentoja ja yhdellä aloitusasennolla voi olla useampia liikkeitä. Tietokantaan täytyy lisätä liitostaulut, joihin merkitään tämä monen suhde moneen-relaatio. Liitostaulun sisältö on kahden toisiinsa liittyvän asian id-numerot.

Liike- ja liikesarjalistat ovat toteutettu yksi moneen -suhteilla (*one-to-many*). Käyttäjällä voi olla monta listaa, mutta lista voi kuulua vain yhdelle käyttäjälle.

### 4.3.5 Koodin generointi Giix-laajennuksella

Yiissä on oma koodi-generaattori, joka on suunniteltu työläiden, mutta usein tarvittavien koodien generoimiseen. Tämä säästää huomattavasti aikaa ja vaivaa. Yiin oma generaattori Gii ei kuitenkaan osaa käsitellä tietokannassa olevia monen suhde moneen relaatioita. Tästä syystä asensin Giix (*Gii Extended*) -laajennuksen, joka osaa käsitellä näitä relaatioita. Näin koodin generointi suoraan tietokantatauluista onnistuu huomattavasti paremmin.

Koodigeneraattori löytyy osoitteesta <http://hostname/index.php?r=gii> Generaattori on oletuksena kommentoitu pois käytöstä. Giin käyttöönoton yhteydessä pitää vielä määrittää salasana, jolla Giitä pääsee käyttämään. Kuvassa 10 näkyy Yiin koodigeneraattorin etusivu.



Kuva 10 Yiin koodigeneraattorin etusivu



Giillä voidaan generoida ohjaintiedostoja, crud-koodia, lomakkeita, mallitiedostoja ja moduuleja. Alimmaisina näkyy Giix-laajennuksen omat malli- ja crudkoodin generaattorit, joita käytin koodin generoimiseen.

## Malli

Malli vastaa tietokannasta hakemiseen ja sinne tallentamiseen liittyvistä asioista. Tämä koodi generoidaan jokaiselle tietokanta taululle erikseen. Giix tarkistaa tietokannasta, mitä tauluja sieltä löytyy, ja ehdottaa sen mukaan, mitä lomakkeeseen kirjoitetaan. Kuvassa 11 näkyy Giixin mallikoodin generointilomake.

The screenshot shows the 'yiisoft/giix code generator' interface. On the left is a sidebar with a 'Generators' menu. The main area is titled 'giix Model Generator' and contains the following fields and options:

- Table Prefix:** tbl\_ (highlighted in yellow)
- Table Name \*:** A text input field containing 'tbl\_trick' with a dropdown menu open. The dropdown lists:
  - tbl\_transition\_tricklist\_assignment
  - tbl\_trick (selected)
  - tbl\_trick\_combo\_assignment
  - tbl\_trick\_setup\_assignment
  - tbl\_trick\_transition\_assignment
  - tbl\_trick\_tricklist\_assignment
  - tbl\_tricklist
- default:** (D:\www\wamp\www\combo\protected\extensions\giix-core\giixModel\templates\default) (highlighted in yellow)
- Preview:** A button at the bottom.

Powered by [Yii Framework](#).  
A product of [Yii Software LLC](#).

Kuva 11 Malli koodin generointi lomake

Mallin generointilomake osaa hakea suoraan tietokannasta taulujen oikeat nimet, joten koodin generointi onnistuu hyvinkin nopeasti. Kuvassa 12 näkyy esitäytetty lomake ja mitä tiedostoja Giix generoi.

yii code generator help | webapp | yii | logout

**Generators**

- Controller Generator
- Crud Generator
- Form Generator
- Model Generator**
- Module Generator
- Bootstrap Generator
- GiixCrud Generator
- GiixModel Generator

## giix Model Generator

This generator generates a model class for the specified database table.  
Fields with \* are required. Click on the highlighted fields to edit them.

**Table Prefix**  
tbl\_

**Table Name \***  
tbl\_trick

**Model Class \***  
Trick

**Base Class \***  
GxActiveRecord

**Model Path \***  
application.models

**Code Template \***  
default (D:\www\wamp\www\combo\protected\extensions\giix-core\giixModel\templates\default)

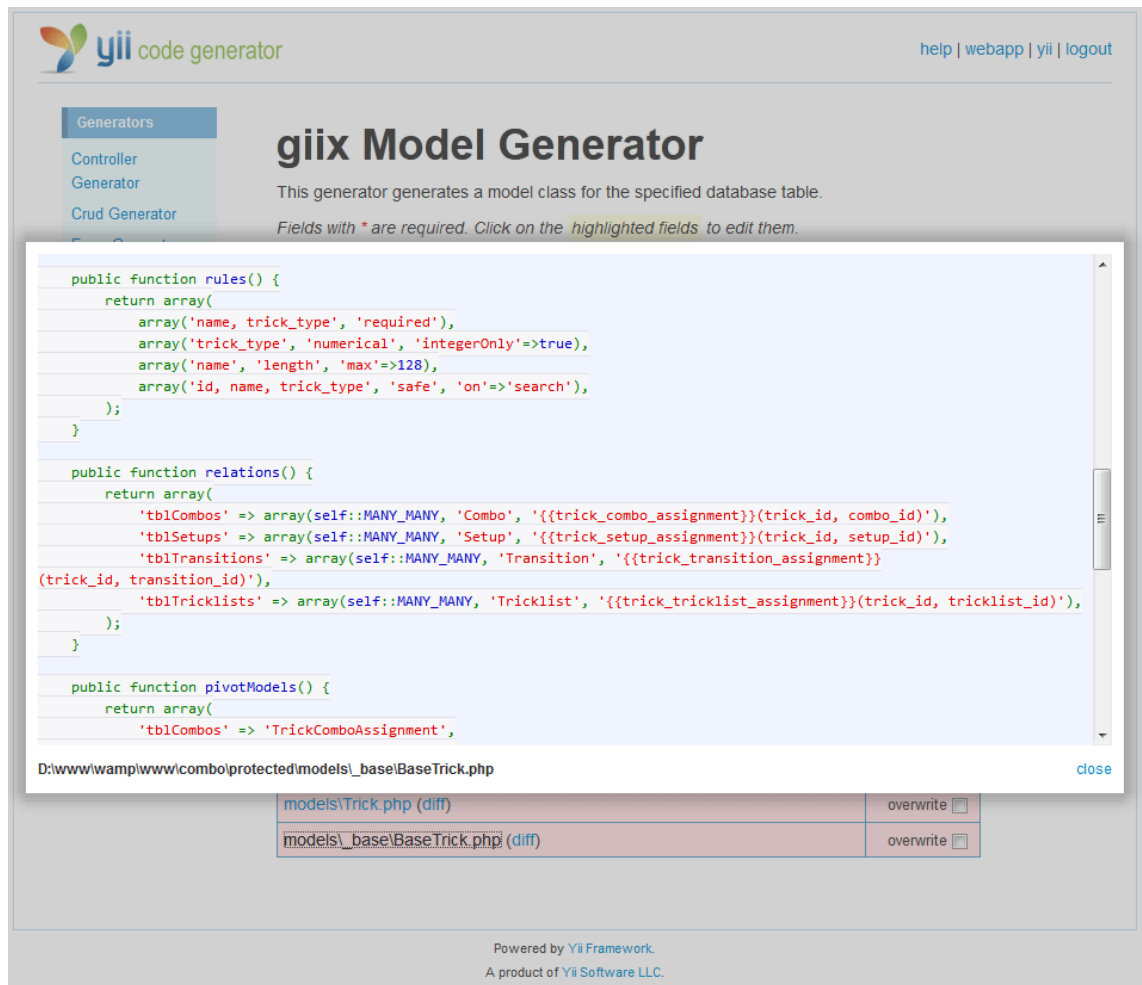
Code File	Generate <input type="checkbox"/>
models\Trick.php (diff)	overwrite <input type="checkbox"/>
models\_base\BaseTrick.php (diff)	overwrite <input type="checkbox"/>

Powered by [Yii Framework](#).  
A product of [Yii Software LLC](#).

Kuva 12 Generaattori näyttää, mitä tiedostoja se generoi, jos lupa annetaan.

Taulun valinnan jälkeen lomake täytetään automaattisesti. Muutoksia voi tehdä, jos oletusasetukset eivät ole sopivia. Giix näyttää, mitä tiedostoja se luo. Tiedoston nimeä klikkaamalla voi nähdä, mitä koodia tiedosto tulee sisältämään (Kuva 13). Jos tiedosto on jo olemassa, näkyy `diff`-linkistä, mitä eroa nykyisen ja generoitavan tiedoston välillä on. `Generate`-painiketta painamalla Giix luo tiedostot niille määrättyyn paikkaan.





yii code generator help | webapp | yii | logout

Generators

- Controller Generator
- Crud Generator

## giix Model Generator

This generator generates a model class for the specified database table.  
Fields with \* are required. Click on the highlighted fields to edit them.

```

public function rules() {
    return array(
        array('name, trick_type', 'required'),
        array('trick_type', 'numerical', 'integerOnly'=>true),
        array('name', 'length', 'max'=>128),
        array('id, name, trick_type', 'safe', 'on'=>'search'),
    );
}

public function relations() {
    return array(
        'tblCombos' => array(self::MANY_MANY, 'Combo', '{{trick_combo_assignment}}(trick_id, combo_id)'),
        'tblSetups' => array(self::MANY_MANY, 'Setup', '{{trick_setup_assignment}}(trick_id, setup_id)'),
        'tblTransitions' => array(self::MANY_MANY, 'Transition', '{{trick_transition_assignment}}(trick_id, transition_id)'),
        'tblTricklists' => array(self::MANY_MANY, 'Tricklist', '{{trick_tricklist_assignment}}(trick_id, tricklist_id)'),
    );
}

public function pivotModels() {
    return array(
        'tblCombos' => 'TrickComboAssignment',
    );
}

```

D:\www\wamp\www\combo\protected\models\\_base\BaseTrick.php close

models\Trick.php (diff)	overwrite <input type="checkbox"/>
models\_base\BaseTrick.php (diff)	overwrite <input type="checkbox"/>

Powered by Yii Framework.  
A product of Yii Software LLC.

Kuva 13 Tiedostoja pääsee tarkastelemaan ennen kuin ne generoidaan.

## CRUD

CRUD eli Create (*luo*), Read (*lue*), Update (*päivitä*), Delete (*poista*). Tämä koodi mahdollistaa esimerkiksi liikkeiden luomisen, lukemisen, päivittämisen ja poistamisen. Koodin generointi toimii hyvin samalla tavalla kuin mallikoodin generointi. Kuvassa 14 näkyy Giixin crud-koodin generointilomake `preview` -painikkeen klikkauksen jälkeen.

yii code generator help | webapp | yii | logout

**Generators**

- Controller Generator
- Crud Generator
- Form Generator
- Model Generator
- Module Generator
- Bootstrap Generator
- GiixCrud Generator
- GiixModel Generator

## giix Crud Generator

This generator generates a controller and views that implement CRUD operations for the specified data model.

*Fields with \* are required. Click on the highlighted fields to edit them.*

**Model Class \***  
Trick

**Controller ID \***  
trick

**Authentication type \***  
Yii access control(more strict ruleset)

**Enable ajax validation \***  
Enable ajax Validation

**Base Controller Class \***  
GxController

**Code Template \***  
default (D:\www\wamp\www\combo\protected\extensions\giix-core\giixCrud\templates\default)

Code File	Generate <input type="checkbox"/>
controllers\TrickController.php (diff)	overwrite <input type="checkbox"/>
views\trick\_form.php (diff)	overwrite <input type="checkbox"/>
views\trick\_search.php (diff)	overwrite <input type="checkbox"/>
views\trick\_view.php (diff)	overwrite <input type="checkbox"/>
views\trick\admin.php (diff)	overwrite <input type="checkbox"/>
views\trick\create.php (diff)	overwrite <input type="checkbox"/>
views\trick\index.php (diff)	overwrite <input type="checkbox"/>
views\trick\update.php	unchanged
views\trick\view.php (diff)	overwrite <input type="checkbox"/>

Powered by Yii Framework.  
A product of Yii Software LLC.

Kuva 14 Kuva Crud generaattorin toiminnasta

Koodin generoinnin lisäksi Giix tarjoaa monta erittäin kätevää funktiota, joilla voi mm. tallentaa monen suhde moneen -relaatioita helposti. Käytin näitä Giixin tarjoamia funktioita koodissani monen suhde moneen -relaatioiden tallentamisessa.

#### 4.3.6 Generaattori

Sivustolla on oma sivunsa liikesarja generaattorille. Tällä sivulla voidaan määrittää asetukset generoitaville liikesarjoille lomakkeen avulla. Liikesarjat tulevat näkyviin lomakkeen viereen, kun generointi-painiketta on painettu. Liikkeiden muokkaaminen ja

tallentaminen tapahtuu myös samalla sivulla. Kaikki kyselyt palvelimelle hoidetaan Ajax-kutsuina, joten sivua ei ole tarpeen ladata uudestaan missään vaiheessa. Tämä nopeuttaa generaattorin käyttöä. Kuvassa 15 on liikesarjojen generointisivu kokonaisuudessaan ja muutama valmiiksi generoitu liikesarja.

The screenshot shows the 'Combo generator' website interface. On the left is a dark sidebar with 'Generator settings'. It includes a 'Tricklist' dropdown menu with options: 'Default tricks', 'Kicks', 'gainer/cork variations', and 'full variations'. Below are 'First trick' and 'Last trick' dropdowns, both set to 'Random'. There are sliders for 'Combo length (max 10)' (set to 3) and 'How many combos (1-10)' (set to 4). A blue 'Generate' button is at the bottom of the sidebar. The main content area is titled 'Compos' and lists four generated combos, each with 'Edit' and 'Save' buttons:

- pop 360 kick - aerial - aerial
- doubleleg - full doubleleg twist s/t swing 900 kick
- aerial - flashkick - raiz
- doubleleg - full doubleleg twist - hyper twist

Kuva 15 Liikesarjojen generointi sivu

Lomakkeella on mahdollista vaikuttaa generoitaviin liikesarjoihin. Aluksi käyttäjä voi määrittää, mitä liikelistaa hän haluaa käyttää. Nämä listat sisältävät ne liikkeet ja transiitiot, joita käyttäjä haluaa generaattorin käyttävän. Näin saadaan liikesarjoista paremmin kaikille sopivia, kun jokainen voi tehdä liikelistansa itse ja lisätä sinne vain liikkeitä, joita itse osaa tai haluaa nähdä generoitavissa liikesarjoissa.

Lomakkeella voidaan määrittää myös ensimmäinen ja viimeinen liike liikesarjasta. Näin käyttäjä saa vaikutettua vieläkin enemmän liikesarjojen sisältöön. Lopuksi käyttäjä voi valita, kuinka pitkiä liikesarjoja hän haluaa, ja kuinka monta liikesarjaa generoidaan.

## Generaattorin logiikka

Generaattorin asetukset lähetetään lomakkeella palvelimelle POST-pyyntönä. Lomakkeen vastaanottaa `GeneratorController`-tiedostossa oleva `actionCombos`-funktio. POST-pyyntön vastaanottaminen ja liikesarjageneraattorin kutsuminen näkyy koodiesimerkissä 2.

```
public function actionCombos()
{
    $settings = Yii::app()->request->getPost('settings');
    .....
    for ($i = 0; $i < $settings['comboCount']; $i++) {
        $thiscombo = Yii::app()->generator->
            generateCombo($settings['length'],$tlist, $settings['firstTrick'],
                $lastTrick);
        .....
    }
}
```

Koodiesimerkki 2 `actionCombos`-funktiossa generaattorin kutsuminen

Ensin funktio hakee lomakkeella POST-metodilla lähetetyt tiedot Yiiin omalla `getPost`-funktioilla. Lomakkeen tiedoista tarkistetaan liikesarjojen määrä ja pituus. Jos lomakkeen tiedot ovat oikein, siirtyy funktio `for`-silmukkaan, joka käydään läpi niin monta kertaa kun liikesarjoja on pyydetty. Silmukassa kutsutaan `generator`-applikaatiokomponentin `generateCombo`-funktioita, joka generoi liikesarjan annettujen parametrien perusteella.

Generaattori on toteutettu applikaatiokomponenttina, jotta sitä voitaisiin käyttää tarvittaessa missä tahansa sivuston sivulla. Liikesarjagenerointi-funktio saa parametreina lomakkeen tiedoista liikesarjan pituuden, valitun liikelistan sekä ensimmäisen ja viimeisen liikkeen.

Generaattorin täytyy ensin määrittää, mitä transitioita voidaan käyttää liikelistassa olevien liikkeiden kanssa liikesarjaa generoidessa. Listassa saattaa olla mukana liike, jolla on sellainen transitio, joka ei käy minkään muun listan liikkeiden kanssa. Mahdollisten transitioiden selvittäminen ennen generoinnin aloittamista on tärkeitä, jotta generaattori ei päätyisi umpikujaan niin helposti. Koodiesimerkissä 3 näkyy `generateCombo`-funktion mahdollisten transitioiden alustaminen.

```

public function generateCombo($length, $tricklist, $firstTrick, $lastTrick)
{
    .....
    foreach ($tricklist->tricks as $t) {
        foreach ($t->transitions as $tra) {
            if (!in_array($tra, $strans)) array_push($strans, $tra);
        }
        foreach ($t->setups as $se) {
            if (!in_array($se, $setu)) array_push($setu, $se);
        }
    }
    $stransitions = array_intersect($setu, $strans);
    $stransitions = array_intersect($stransitions, $tricklist->transitions);
    .....
}

```

### Koodiesimerkki 3 Mahdollisten transitioiden selvittäminen

Koodi käy läpi liikelistan liikkeet, ja ottaa transitiot ja setupit eli aloitus- ja lopetusliikkeit omiin taulukkoihinsa ja karsii näistä päällekkäiset merkinnät pois. Lopuksi funktio karsii vielä ne transitiot, joita ei ole liikelistassa määritelty. Tässä vaiheessa täytyy muistaa, että aloitusliikkeelle (*setup*) löytyy aina vastaavan niminen lopetusliike (*transitio*) ja liikkeiden välissä oleva aloitus- ja lopetusliikkeen yhdistelmä on aina nimeltään transitio. Kun generoitavaan liikesarjaan sopivat transitiot ovat selvillä, siirtyy funktio koodiesimerkissä 4 olevaan liikesarjan generointia ohjaavaan for-silmukkaan.

```

for ($i = 1; $i < $length; $i++) {
    if ($i == 1) {
        if ($firstTrick) {
            array_push($combo, Trick::model()->findByPk($firstTrick));
            $miniCombo = $this->generateMiniCombo($tricklist, $stransitions,
end($combo));
        } else {
            $miniCombo = $this->generateMiniCombo($tricklist, $stransitions,
end($combo));
            array_push($combo, $miniCombo[0]);
        }
    }
    .....
    else {
        $miniCombo = $this->generateMiniCombo($tricklist, $stransitions,
end($combo));
    }
    array_push($combo, $miniCombo[1]);
    array_push($combo, $miniCombo[2]);
}

```

### Koodiesimerkki 4 liikesarjan ensimmäisen liikkeen valinta for-silmukassa

Liikesarjan generointi aloitetaan ensimmäisen liikkeen lisäämisellä liikesarjaan. Jos ensimmäinen liike on määritelty lomakkeessa, haetaan se tietokannasta, ja lisätään liike-

sarjan ensimmäiseksi liikkeeksi. Muuten ensimmäisen liike arvotaan satunnaisesti `generateMiniCombo`-funktiolla.

`GenerateMiniCombo`-funktio palauttaa aina lyhyitä liike-transitio-liike-liikesarjoja. Liikesarjat kasataan generoimalla näitä lyhyitä liikesarjoja peräkkäin niin monta kertaa, että liikesarjan pituus täyttää vaaditun pituuden.

Generaattorin ensimmäisellä kierroksella lisätään kaikki liikkeet, jotka `generateMiniCombo` palauttaa, mutta seuraavilla kierroksilla funktion palauttamasta liikesarjasta käytetään vain transitio ja viimeinen liike, sillä sen ensimmäinen liike on liikesarjan tämän hetkinen viimeinen liike ja sitä tarvitaan, jotta liikesarjan seuraava liike generoituisi oikein. Ensimmäisellä kierroksella liikesarja kasvaa siis kahdella liikkeellä, ja sitä seuraavilla kierroksilla liikesarjaan lisätään aina vain yksi transitio ja yksi liike, jotka arvotaan koodiesimerkissä 4 alhaalla olevan `else`-osion koodilla.

Koodiesimerkissä 4 pisteiden kohdalla on poikkeussäännöt liikesarjan viimeisille kierroksille, mutta selkeyden vuoksi kerron niistä hieman myöhemmin.

Koodiesimerkissä 5 näkyy `generateMiniCombo`-funktion ensimmäisen liikkeen hakeminen. Tämä suoritetaan vain, jos `generateMiniCombo` ei saa parametrina `$prevTrick` muuttujaa eli lyhyen liikesarjan ensimmäistä liikettä. Tässä tilanteessa tämä suoritettaisiin vain, jos `generateCombo`-funktio on ensimmäisellä kierroksella, ja ensimmäistä liikettä ei ole annettu lomakkeella.

```

public function generateMiniCombo($tricklist, $transitions, $prevTrick =
null, $lastRound = false)
{
.....
if (!$prevTrick) {
    do {
        $prevOk = false;
        $prevTrick = $tricklist->tricks[array_rand($tricklist->tricks)];

        if (array_intersect($prevTrick->transitions, $transitions)) {
            $prevOk = true;
        }
    } while ($prevOk == false);
}
$miniCombo[0] = $prevTrick;
.....
}

```

### Koodiesimerkki 5 Ensimmäisen liikkeen hakeminen

Ensin generateMiniCombo-funktio tarkistaa, onko sille välitetty liikesarjan ensimmäistä liikettä. Jos ensimmäistä liikettä ei ole asetettu, hakee generateMiniCombo-funktio liikelistasta sellaisen liikkeen, jonka jokin transiioista löytyy aikaisemmin määritellyistä mahdolliset transiitot -taulukosta. Koodiesimerkissä 6 näkyy seuraavan transiition ja liikkeen valinta.

```

public function generateMiniCombo($tricklist, $transitions, $prevTrick =
null, $lastRound = false)
{
    .....
    $possibleTransitions = array_intersect($transitions, $prevTrick-
>transitions);

    if ($possibleTransitions) {
        $trickOk = false;
        do {
            $transition = $possibleTransitions[array_rand($possibleTransitions)];

            foreach ($tricklist->tricks as $tri) {
                foreach ($tri->setups as $setup) {
                    if ($setup->name == $transition->name) {
                        if ($lastRound) {
                            if (!in_array($tri, $possibleTricks))
                                array_push($possibleTricks, $tri);
                        } else {
                            if (array_intersect($tri->transitions, $transitions)) {
                                if (!in_array($tri, $possibleTricks))
                                    array_push($possibleTricks, $tri);
                            }
                        }
                    }
                }
            }
        } while ($trickOk == false);
        .....
    }
}

```

### Koodiesimerkki 6 Seuraavan transition ja liikkeen generointi

Jos mahdollisia transitoita on löytynyt, siirtyy funktio do-while-silmukkaan. Tässä sil-  
mukassa funktio valitsee satunnaisesti jonkun mahdollisista transiatioista. Funktio käy  
kaikki liikelistan liikkeitä ja niiden setup-arvot eli transiitot, joista liike voidaan tehdä, ja  
vertaa niitä aikaisemman liikkeen transiioon. Eli funktio vertailee aikaisemman liik-  
keen lopetusasentoa muiden liikkeiden aloitusasentoihin. Kaikki liikkeit, jotka sopivat  
aikaisemman liikkeen kanssa yhteen valittua transiitiota käyttäen, sijoitetaan mahdolliset  
liikkeet -taulukkoon. Ennen sijoittamista tarkistetaan kuitenkin vielä, että valitulla uu-  
della liikkeellä on sellainen transiio, joka käy muiden liikelistan liikkeiden kanssa. Jos  
tätä ei tarkisteta, voi seuraavalla kierroksella käydä niin, ettei yksikään liike sovi tämän  
liikkeen jälkeen. Jos yhtään liikettä ei löydy, arpoo funktio uuden transition ja yrittää  
uudelleen. Näin voidaan tehdä, koska generateCombo-funktio karsi aikaisemmin pois  
kaikki ne transiitot, joita ei voida käyttää liikesarjassa. Eli sopiva pari löytyy varmasti.  
Seuraavassa koodiesimerkissä 7 funktio palauttaa generoimansa lyhyen liikesarjan.



```

public function generateMiniCombo($tricklist, $transitions, $prevTrick =
null, $lastRound = false)
{
.....
if ($possibleTricks) {
    $miniCombo[1] = $transition;
    $miniCombo[2] = $possibleTricks[array_rand($possibleTricks)];

    return $miniCombo;
}
.....
}

```

Koodiesimerkki 7 Lyhyen liikesarjan palautus

Kun liikkeitä on löytynyt, sijoittaa funktio transition palautettavaan taulukkoon ja arpoo satunnaisesti viimeisen liikkeen palautettavan taulukon viimeiseksi.

Generaattorin viimeinen kierros tarvitsee hieman erilaiset säännöt kuin ensimmäinen ja kaikki aikaisemmat kierrokset. Koodiesimerkkissä 8 näkyy poikkeussäännöt viimeisille kierroksille.

```

else if (($i + 2 == $length) && ($lastTrick)) {
    $miniCombo = $this->generateSecondLastTrick($tricklist, $transitions,
end($combo), $lastTrick);
}
else if (($i + 1 == $length) && ($lastTrick)) {
    $miniCombo = $this->generateLastTrick($tricklist, $transitions,
end($combo), $lastTrick);
}
else if (($i + 1 == $length) && (!$lastTrick)) {
    $miniCombo = $this->generateMiniCombo($tricklist, $transitions,
end($combo), true);
}

```

Koodiesimerkki 8 Liikesarjan poikkeussääntöjä viimeisille kierroksille

Jos käyttäjä on valinnut viimeisen liikkeen lomakkeella, katsoo generaattori toiseksi viimeisellä kierroksella liikkeen, joka sopii sitä edeltävän liikkeen ja annetun viimeisen liikkeen väliin. Jos sopivaa liikettä ei löydy, näytetään käyttäjälle virhe. Viimeisellä kierroksella generaattori vielä arpoo sopivan transition toiseksi viimeisen ja annetun viimeisen liikkeen väliin.

Jos viimeistä liikettä ei ole annettu lomakkeella, arpoo generaattori viimeisen liikkeen muuten normaalisti, mutta ei tarkista enää viimeisen liikkeen transitioita. Eli liikesarjan loppuun voi tulla myös liike, josta generaattori ei voisi enää jatkaa liikkeen generointia

annetun liikelistan liikkeillä. Tämä näkyy aikaisemmassa koodiesimerkissä 6 kohdassa, jossa tarkistetaan `$lastRound`-muuttuja.

Kun liikesarjaan on generoitu tarpeeksi liikkeitä, lopettaa funktio `for`-silmukan läpi käymisen ja palauttaa taulukon, joka sisältää liikkeitä ja transiitioita. Tämä taulukko lisätään `actionCombos`-funktiossa taulukkoon, joka sisältää liikesarjoja. Liikesarjat identifioidaan sivustolla niiden nimien perusteella. Liikesarjojen nimet myös määrittävät liikkeiden ja transiitoiden järjestyksen liikesarjassa. Koodiesimerkissä 9 näkyy, miten liikesarjojen nimet muotoillaan. Nimissä liikkeiden ja transiitoiden nimet erotellaan pilkulla.

```
public function actionCombos()
{
    .....
    for ($i = 0; $i < $settings['comboCount']; $i++) {
        $thiscombo = Yii::app()->generator->generateCombo($settings['length'],
        $tlist, $settings['firstTrick'], $lastTrick);
        $combos[$i] = $thiscombo;

        foreach ($thiscombo as $trick) {
            if ($comboName === "") {
                $comboName = $trick->name;
            } else {
                $comboName .= "," . $trick->name;
            }
        }
        $comboNames[$i]["name"] = $comboName;
        $comboName = "";
    }
}

Yii::app()->clientScript->scriptMap['*.js'] = false;
Yii::app()->clientScript->scriptMap['*.css'] = false;

$this->renderPartial('_combos', array('combos' => $combos, 'comboNames' =>
$comboNames), false, true);
}
```

### Koodiesimerkki 9 Liikesarjojen nimien muotoilu

Lopuksi, kun liikesarjat on generoitu ja niiden nimet kirjoitettu oikeaan muotoon, luo funktio osittaisen näkymän `renderPartial`-funktiolla. Näkymään lähetetään parametreina generoidut liikesarjat ja niiden nimet. Ennen näkymän luomista `scriptMap`-taulukosta poistetaan kaikki JavaScript- ja CSS-tiedostot, koska käytössä on Ajax-kutsu. Jos näitä ei poistettaisi, lisäisi Yii sivulle Ajax-kutsun jälkeen kaikki tiedostot uudel-

leen. Silloin esimerkiksi jQuery olisi sivulla kahteen kertaan, ja se aiheuttaisi paljon ongelmia.

Näkymässä tulostetaan liikesarjat näkyville, ja jokaisen liikesarjan viereen lisätään muokkaus- ja tallennus-painike. Liikesarjojen nimet käännetään vielä käyttäjän määrittelemien aliasten mukaan. Kaikkia liikkeitä ja liikesarjoja käsitellään koodissa niiden raakanimillä. Liikesarjat, joissa on käytetty raakanimiä, eivät ole kovin helppolukuisia. Nimien kääntämistä varten on luotu alias applikaatiokomponentti, jota käytetään aina, kun halutaan näyttää liike tai liikesarja sivustolla käyttäjän määrittelemässä luettavammassa muodossa.

### Liikesarjan muokkaaminen ja tallentaminen

Generaattorista tulleita liikesarjoja voi muokata saman tien. Otetaan esimerkiksi kuvassa 16 näkyvä liikesarja pop 360 kick – aerial – aerial. Tässä liikesarjassa on hyvä alku, mutta viimeinen liike voisi olla jotain muuta.

## Combos

pop 360 kick - aerial - aerial

Edit

Save

Kuva 16 Liikesarja jota käyttäjä haluaa muokata

Liikesarja voisi olla mieluummin pop 360 kick – aerial – hyper twist. Liikesarjan muokkaustilaan pääsee painamalla Edit-painiketta. Liikesarjan muokkaamisessa käytetään tietokannasta löytyviä raakanimiä. Tämä johtuu siitä, että käyttäjä näkee transitiot niiden oikeilla nimillä ja liikesarjan editointi tapahtuu mahdollisimman tarkasti. Muokkaustila näkyy kuvassa 17.

## Combos

pop 360 kick absetup aerial absetup aerial

Add new trick

Done

Kuva 17 Liikesarja muokkaustilassa

Liikesarjojen muokkaaminen on toteutettu widgettinä, sillä tätä samaa toiminnallisuutta tarvitaan mm. erillisellä liikesarjojen muokkaussivulla. Liikesarjan editointiin tarvittava widget haetaan Ajax-kutsuna. Sivulla on kuuntelija, joka huomaa, kun Edit-painiketta on painettu (Koodiesimerkki 10).

```
Yii::app()->clientScript->registerScript("generaattoriNapit" . $editUniq, "  

$('#" . $editUniq . "').click(function() {  

    var comboName = $(this).siblings('input').val();  

    var update = $(this).closest('.comboRow');  

    var url = '" . Yii::app()->createUrl('generator/editCombo') . "' +  

    '?comboName='+comboName;  

    editCombo(url, update);  

});  

");
```

Koodiesimerkki 10 Sivulle upotettu Edit-painikkeen kuuntelija

Painikkeen kuuntelija kerää tarvittavat tiedot liikesarjasta, ja ne välitetään parametrina Ajax-funktiolle. Liikesarjan nimi, joka liitetään osoitteeseen, jota kutsutaan Ajax-funktiolla, ja minkä liikesarjan kohdalle editointi-widget liitetään. Lopuksi kuuntelija kutsuu funktiota, joka suorittaa Ajax-kutsun. Ajax kutsun suorittava funktio näkyy koodiesimerkissä 11.

```
function editCombo(url, update) {
  jQuery.ajax({
    'url': url,
    'data': $(this).serialize(),
    'type': 'post',
    'dataType': 'json',
    'success': function (data) {
      update.html(data.div);
    }, 'cache': false});
  return false;
}
```

### Koodiesimerkki 11 Ajax-kutsun suorittaminen

Ajax-kutsu suoritetaan käyttämällä jQueryn Ajax-funktiota. Ajax-kutsun vastaanottaa generator-ohjaimessa oleva `actionEditCombo`-toiminto. Tämä toiminto luo osittaisen näkymän `_editCombo`-tiedostosta. Osittaiseen näkymään lähetetään liikesarjan nimi (Koodiesimerkki 12).

```
public function actionEditCombo($comboName)
{
  .....
  echo CJSON::encode(array(
    'status' => 'success',
    'div' => $this->renderPartial('_editCombo', array('comboName' =>
$comboName), true, true)
  ));
  exit;
}
```

### Koodiesimerkki 12 Ajax-kutsun käsittelevä toiminto

Näkymässä `_editCombo` kutsutaan liikesarjan editointi-widgettiä. Widget saa parametrina liikesarjan nimen (Koodiesimerkki 13). Widgetin jälkeen näkymään lisätään vielä Done-painike, jota käytetään, kun halutaan lopettaa liikesarjan editointi.

```
$this->widget('ext.edit-combo.editCombo', array(
  'comboName' => $comboName,
));
```

### Koodiesimerkki 13 Liikesarjan editointi-widgetin kutsuminen

Liikesarjan editointiin tarvittava widget käynnistetään, ja se luo oman näkymänsä (Koodiesimerkki 14). Widgetin omassa näkymässä on mukana kaikki tarvittavat painikkeet ja JavaScript-koodit, joita liikkeen muokkaamiseen tarvitaan.

```

class EditCombo extends CWidget
{
    public $comboName;

    public function run()
    {
        $this->render("editCombo", array('comboName'=> $this->comboName));
    }
}

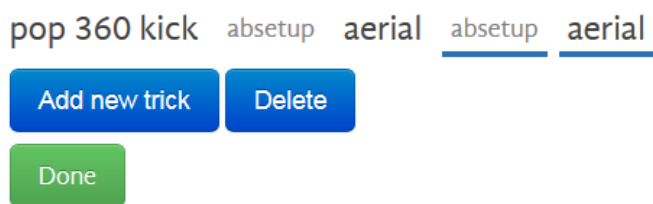
```

#### Koodiesimerkki 14 Liikesarjan editointi-widgetin käynnistäminen

Kun kaikki näkymät on luotu, palataan takaisin `actionEditCombo`-toimintoon, joka palauttaa JSON-muodossa osittaiset näkymät Ajax-kutsun suorittaneelle funktiolle. Tämä näkyy koodiesimerkissä 12. Koodiesimerkissä 11 näkyy vielä rivi `update.html(data.div);`, joka päivittää liikesarjan editointi-widgetin sivulle käyttäjän nähtäville.

Liikesarjan muokkaustilassa käyttäjä voi poistaa liikkeitä liikesarjasta sekä lisätä siihen uusia liikkeitä. Liikkeen poistaminen onnistuu, kun valitaan ensin poistettavat liikkeet klikkaamalla liikkeen tai transition nimeä, ja sen jälkeen painetaan `Delete`-painiketta. `Delete`-painike näkyy vain silloin, kun joku liikesarjan osa on valittuna. Poistaminen näkyy kuvassa 18.

## Combos



Kuva 18 Liikesarjan osien poistaminen

Liikesarjasta poistamisen lisäksi on mahdollista lisätä uusia liikkeitä. Tämä onnistuu `Add new trick` -painikkeella. (Kuva 19)

## Combos

pop 360 kick absetup aerial

Kuva 19 Uuden liikkeen lisääminen liikesarjaan

Liikkeiden ja transitioiden lisääminen liikesarjaan käy helposti. Uuden liikkeen lisäämisessä generaattori ehdottaa viimeisen liikkeen jälkeen sopivia transitoita, jos liikesarjan viimeinen osa on transiio niin generaattori ehdottaa sen transition jälkeen sopivia liikkeitä. Koska liikkeet ja transitiot liittyvät toisiinsa, voidaan nämä mahdolliset liitokset hakea suoraan tietokannasta katsomalla liikkeen ja transition yhdistävää liitostaulua.

Kun jotain harmaista painikkeista painetaan, lisätään se liikesarjan loppuun (Kuva 20).

## Combos

pop 360 kick absetup aerial absetup

Kuva 20 Transiio lisätty liikesarjaan

Absetup-transitiota on nyt painettu, ja se löytyy liikesarjan lopusta. Tämän jälkeen generaattori hakee tietokannasta ne liikkeet, joihin on liitettyä absetup aloitusasennoksi. Käyttäjä voi valita myös satunnaisesti jonkun painamalla `Random`-painiketta. Vaihtoeh-

toisesti liikkeen nimi tai osa liikkeen nimestä voidaan kirjoittaa tekstikenttään. Tämä suodattaa hakutuloksista pois ne liikkeet tai transitiot, jotka eivät sovi kirjoitettuun hakusanaan (Kuva 21).

## Combos

pop 360 kick absetup aerial absetup

The screenshot shows a web interface for editing a 'Combos' list. At the top, the current list is 'pop 360 kick absetup aerial absetup'. Below this is a blue 'Add new trick' button. A search input field contains the text 'hyper', and a blue 'Random' button is to its right. A dropdown menu is open below the search field, listing the following items: 'hyper twist' (highlighted), 'hyperfull', 'hyperfull hook', and 'hyperhook'. To the right of the dropdown, two grey buttons are visible: 'hyperfull hook' and 'hyperhook'.

Kuva 21 Hakutulosten suodattaminen

Tarkoituksena oli saada liikesarjan viimeinen liike vaihdettua hyper twist -liikkeeksi. Tekstikenttään voi kirjoittaa ”hyper” ja tuloksista suodatetaan kaikki ne liikkeet, joiden nimessä lukee ”hyper”. Uuden liikkeen lisäämisen jälkeen liikesarja on tältä erää valmis (Kuva 22).

## Combos

pop 360 kick absetup aerial absetup hyper twist

The screenshot shows the 'Combos' interface after editing. The current list is 'pop 360 kick absetup aerial absetup hyper twist'. Below the list is a blue 'Add new trick' button. A search input field contains the text 'Add transition', and a blue 'Random' button is to its right. Below the search field, a row of grey buttons contains the following items: 'absetup', 'carry through', 'cheat', 'pop', 'standing', and 'wrap through'. At the bottom left, there is a green 'Done' button.

Kuva 22 Halutunlaiseksi muokattu liikesarja

Liikesarjan editoinnin voi lopettaa painamalla `Done`-painiketta. Painikkeen painamisen jälkeen suoritetaan hyvin samantapaiset Ajax-kutsut kuin liikesarjan editointi-widgetin hakemisessa, mutta tällä kertaa ei ladata widgettiä ja kutsun vastaanottaa generator-ohjaimessa eri funktio. Tämä funktio luo osittaisen näkymän, joka näyttää samalta kuin se näkymä, mistä alun perin painettiin `Edit`-painiketta. Sivulla näkyy nyt äskettäin edi-



toitu liikesarja (Kuva 23). Huomaa, että liikesarjan liikkeiden nimet on taas käännetty käyttäjän itse määrittelemiin nimiin alias applikaatiokomponentin avulla.

## Combos

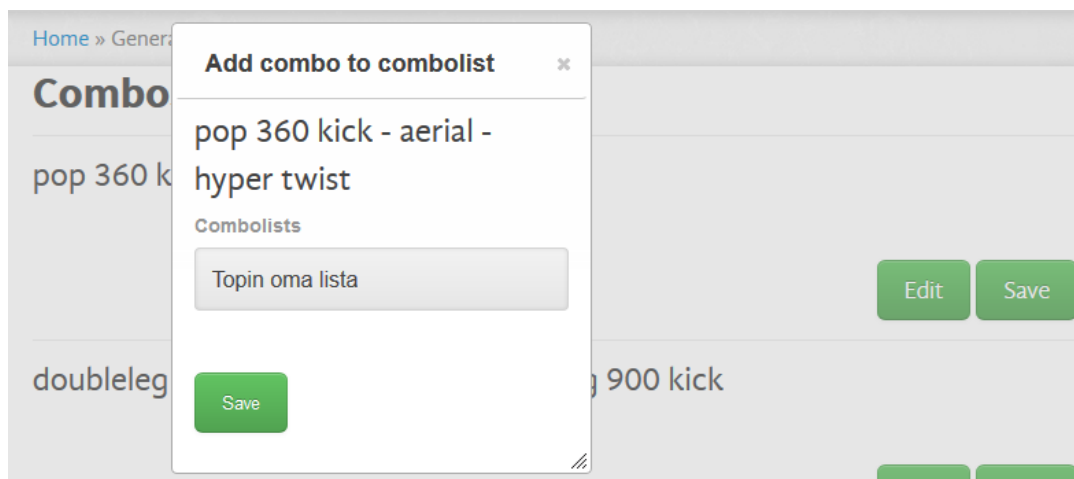
pop 360 kick - aerial - hyper twist

Edit

Save

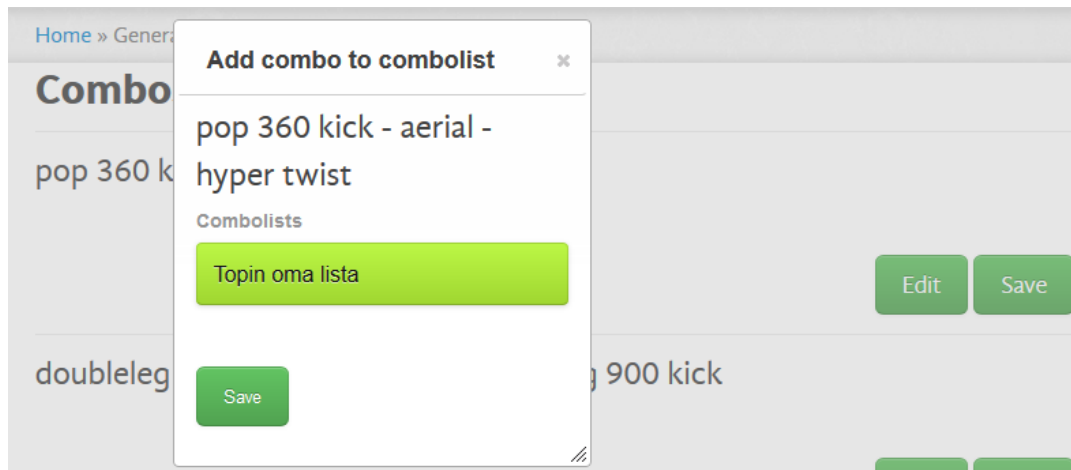
Kuva 23 Liikesarja muokkaamisen jälkeen

Liikesarja voidaan tallentaa ennen tai jälkeen editoinnin. Tallentaminen tapahtuu painamalla `Save`-painiketta (Kuva24).



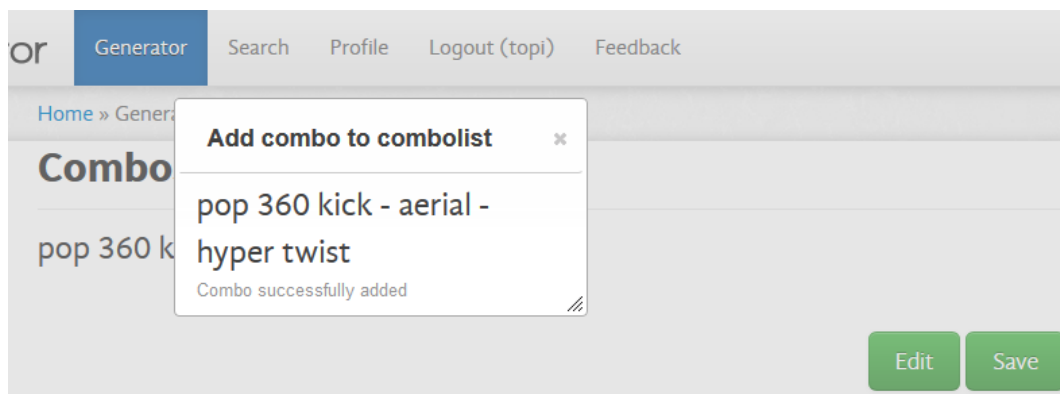
Kuva 24 Liikesarjan tallentaminen

`Save`-painikkeen painaminen avaa sivulle uuden ikkunan. Tässä ikkunassa näkyvät käyttäjän kaikki liikesarjalistat. Listoista valitaan ne listat, joihin haluaa liikesarjan tallentaa. Koska generaattorista tulee satunnaisia liikesarjoja, saattaa liikesarja olla jo jossain käyttäjän listoissa. Silloin se lista näytetään jo valmiiksi aktiivisena ja käyttäjä voi halutessaan myös poistaa tämän liikesarjan listastaan tässä vaiheessa (Kuva 25).



Kuva 25 Listan valinta liikesarjan tallentamisessa

Kun liikesarja tallennetaan, näytetään käyttäjälle ilmoitus liikesarjan tallentamisesta ja ikkuna sulkeutuu itsestään (Kuva 26).



Kuva 26 Liikesarjan tallentaminen suoritettu onnistuneesti

Kuvassa 27 on generaattori-sivu vielä tallentamisen jälkeen.

The screenshot shows the 'Combogenerator' website interface. On the left is a dark sidebar titled 'Generator settings' containing:
 

- Tricklist:** A dropdown menu with options: Default tricks, Kicks, gainer/cork variations, and full variations.
- First trick:** A dropdown menu set to 'Random'.
- Last trick:** A dropdown menu set to 'Random'.
- Combo length (max 10):** A slider and input field set to '3'.
- How many combos (1-10):** A slider and input field set to '4'.
- Generate:** A blue button at the bottom of the sidebar.

 The main content area is titled 'Compos' and shows a list of generated combos:
 

- 'pop 360 kick - aerial - hyper twist' with 'Edit' and 'Save' buttons.
- 'doubleleg - full doubleleg twist s/t swing 900 kick' with 'Edit' and 'Save' buttons.
- 'aerial - flashkick - raiz' with 'Edit' and 'Save' buttons.
- 'doubleleg - full doubleleg twist - hyper twist' with 'Edit' and 'Save' buttons.

 The top navigation bar includes 'Generator', 'Search', 'Profile', 'Logout (topi)', and 'Feedback'. A breadcrumb trail shows 'Home > Generator'.

Kuva 27 Generointi-sivu tallennuksen jälkeen

Seuraavaksi käyttäjä voi joko muokata liikesarjaa uudestaan tai muokata muita liikesarjoja tai halutessaan generoida täysin uudet liikesarjat.

#### 4.4 Käyttäjien hallinta

Toteutin käyttäjien hallinnan käyttämällä Yii-user-laajennusta. Yii-userin ominaisuuksiin kuuluu kattava paketti käyttäjien hallintaan tarvittavia asioita. Laajennuksen tärkeimpiin ominaisuuksiin kuuluu rekisteröinti, kirjautuminen, käyttäjäprofiili-sivut, ja käyttäjien hallinta.

Käytin käyttäjien hallintaan valmista laajennusta, sillä halusin keskittyä enemmän sivuston muihin kohtiin, kuten generaattoriin. Laajennusta käyttämällä on sivustolla varmasti myös vähemmän virheitä ja se on turvallisempi käyttää, kun koodi on Yii-yhteisön hyväksymää ja testattu kunnolla.

Yii-user on oikeastaan moduuli. Moduuli on hieman suurempi kokonaisuus kuin laajennus. Moduulit eroavat laajennuksista siten, että moduulit ovat tavallaan aliohjelmiä. Niillä on omia ohjaimia, malleja ja näkymiä. Yii-user toteuttaa myös omat tietokantataulut asennuksen yhteydessä.

Salasanojen salaaminen ei ollut sillä tasolla, mitä halusin niiden olevan. Yii-user laajennuksessa on mahdollista valita joko md5- tai sha1-salaus. Molemmat ovat helposti murrettavissa sateenkaaritaulujen avulla. Molemmat tavat tuottavat aina samasta sanasta samannäköisen sarjan merkkejä. Sateenkaaritauluissa on nämä merkkisarjat valmiina jokaisesta mahdollisesta sanasta, jolloin niistä on helppo etsiä vastaavat parit ja näin murtaa salasanat.

Salasanojen salaamiseen (*Password hashing*) on hyvä nyrkkisääntö. Älä koskaan keksi salausrakenteita itse! Ellei tietysti ole alan asiantuntija. Salasanojen salaamiseen on kuitenkin jo erittäin hyviä valmiita algoritmeja. Käytin Combogeneraattorissa phpass nimistä laajennusta. Laajennus käyttää blowfish-algoritmia. Näin salasanat pysyvät paremmin turvassa.

### **Käyttäjien oikeudet**

Yii:ssä on oma autentikointijärjestelmänsä. Käytin apuna Auth-nimistä laajennusta. Auth antaa graafisen käyttöliittymän autentikointi-itemien hallintaan ja helpottaa oikeuksien jakamista sekä ylläpitämistä.

Yii:ssä on käytössä RBAC (*Role-Based Access Control*) -tekniikka. (Xue 2012, 196) RBAC-tekniikassa käyttäjille jaetaan rooleja (*role*). Eri rooleille voidaan sallia suoritettavia erilaisia tehtäviä (*task*) ja nämä tehtävät koostuvat erilaisista toiminnoista (*operation*).

Aluksi tietokantaan lisätään taulut `AuthAssignment`, `AuthItemChild` ja `AuthItem`, joihin tallennetaan oikeuksiin liittyviä tietoja. Taulujen skeemat löytyvät Yii:n framework-kansiosta.

Auth-laajennuksella on graafinen käyttöliittymä sivuston sivulla, johon vain erikseen ylläpitäjäksi määritelty käyttäjä pääsee. Tämä helpottaa käyttöä, kun käyttäjien oikeuk-

sia ei tarvitse määrittellä ohjelmakoodissa. Käyttöliittymän kautta onnistuu uusien roolien, tehtävien ja operaatioiden lisääminen sekä muokkaaminen. Kun tarvittavat tiedot on tallennettu tietokantaan, voi ohjelmakoodiin lisätä tarkistuksia kyseisen käyttäjän oikeuksista. Koodiesimerkissä 15 tarkistetaan, voiko käyttäjä muokata liikesarjalistaa. Jos käyttäjä on itse luonut listan, sallitaan muokkaaminen. Muuten näytetään virheilmoitus.

```
if (Yii::app()->user->checkAccess('combolist.update',
    array('combolist'=>$model)))
{
    .....
}
else
{
    throw new CHttpException(401, 'Access denied.');
```

Koodiesimerkki 15 Käyttäjän oikeuden tarkistaminen

Auth voidaan suorittaa myös suodattimena. Suodatin (*filter*) on ohjelmakoodia, joka suoritetaan ennen tai jälkeen toiminto-funktion. (Xue 2012, 31) Suodattimilla voidaan määrittää, suoritetaanko toiminto vai ei. Koodiesimerkissä 16 on suodatettu kaikki muut toiminnot, eli sivut, paitsi index ja view. Tätä suodatinta käytetään mm. liikkeiden, transiitoiden, liikesarjojen, liikelistojen ja liikesarjalistojen ohjain-tiedostoissa.

```
public function filters()
{
    return array(
        array('auth.filters.AuthFilter - index, view'),
    );
}
```

Koodiesimerkki 16 Auth-lisäosan käyttö suodattimena

Auth-lisäosa antaa siis joustavan ja nopean tavan hallita käyttäjenoikeuksia. Tulevaisuudessa, kun sivustolle lisätään kommentit, voidaan Auth-lisäosalla luoda uusi rooli, moderaattorit. Moderaattoreille voidaan antaa oikeudet kaikkien käyttäjien kommenttien muokkaamiseen ja poistamiseen.

### Uuden käyttäjän alustaminen

Rekisteröitymisen yhteydessä käyttäjälle luodaan alkutiedot. Alkutietojen tarkoituksena on helpottaa sivuston käyttöönottamista. Käyttäjälle kopioidaan liikkeiden nimien ly-

henteet eli aliakset. Samat, jotka näkyvät vierailijoille. Käyttäjälle luodaan myös valmiiksi tyhjat liike ja liikesarjalistat.

#### 4.5 Sivuston testaaminen

Sivustoa ohjelmoidessani olen testannut sitä suosituimmilla moderneilla selaimilla, kuten Mozilla Firefox, Google Chrome, Opera, Internet Explorer ja Safari. Selaimista minulla on ollut käytössä aina uusin versio.

Sivuston testaaminen älypuhelimilla ja tablet-tietokoneilla on tehty pääosin Operan mobiili-emulaattorilla. Välillä, kun se on ollut mahdollista, on sivustoa testattu loppukäyttäjän omistamalla älypuhelimella tai tabletilla.

Näiden lisäksi olen tietysti testannut sivustoa loppukäyttäjillä. (Kuva 28)



Kuva 28 Sivuston testaamista

Sivustoa ja generaattorin toimintaa on testattu tulevien loppukäyttäjien toimesta usean kerran. Testikäyttäjille kuuluu myös kiitos monien uusien liikkeiden lisäämisestä sivus-

tolle. Testaus on suoritettu triikkaajille luontaisessa ympäristössä eli harjoitusten aikana voimisteluhallissa. Näin myös generaattorin arpomia liikesarjoja on päässyt heti kokeilemaan.

## 5 VALMIIN TYÖN ARVIOINTI

Combogeneraattori täyttää suurimman osan projektin alussa määrittämistäni vaatimuksista. Kuitenkin nälkä kasvaa syödessä ja lisäsin sivustoon aina uusia asioita, kun vanhat alkoivat toimia. Lopulta jouduin siirtämään myöhemmälle monta hyvää ja toteutuskelpoista ideaa. Tästä syystä jäi hieman sellainen olo, että sivusto jäi kesken. Haluankin vielä kehittää sivustoa pidemmälle ennen kuin julkaisen sen suuremmalle yleisölle. Joka tarkoittaa ulkomaalaisia triikkaus-aiheisia keskustelufoorumeita ja laajempaa levittämistä sosiaalisenmedian kautta.

Olen erittäin tyytyväinen sivustoon jo nyt, vaikka osa asioista joutuukin vielä odottamaan toteuttamista. Opinnäytetyötä aloittaessani minulla ei ollut aavistustakaan, kuinka saisin tämän projektin valmiiksi. Kuitenkin työn edetessä ongelmat ratkesivat yksi kerrallaan ja sain sivustosta jo sellaisen version jota voi käyttää.

Opinnäytetyöraportin kirjoittaminenkin meni hyvin. Sain mielestäni hyvän rakenteen työlleni ja käsittelin laajasti sivuston kehittämistä aina suunnittelusta valmiiseen sivustoon asti. Aluksi oli tarkoitus käyttää ja kirjoittaa myös testivetoisesta kehityksestä ja esitellä, kuinka versionhallintaa voidaan käyttää WWW-sivusto projekteissa, mutta työssä alkoi olla jo liikaa asioita, ja minun oli pakko rajata työtäni uudelleen. Rajaaminen oli hyvä ratkaisu, sillä kerkesin paremmin keskittymään muihin asioihin.



## 6 SIVUSTON JATKOKEHITYS

Tulen varmasti jatkamaan sivuston kehittämistä vielä myöhemminkin. Jouduin rajamaan paljon hyviä ideoita pois koska aika ei riittänyt kaikkeen. Jatkokehitys jakautuu kahteen osaan. Ne, jotka ovat jo suunnitteluvaiheessa, ja jotka aion varmasti toteuttaa, sekä ne, jotka ovat vielä ideoita, enkä ole vielä päättänyt toteutetaanko niitä.

### **Toteutettavat ominaisuudet**

Generaattorista voidaan nyt generoida loogisia liikesarjoja, jotka ovat mahdollisia toteuttaa. Generaattori ei kuitenkaan ymmärrä, että useat liikesarjat ovat todella vaikeita. Tästä syystä toteutan generaattoriin vaikeusastepisteet. Näillä pisteillä voidaan määrittää, kuinka vaikea tietty liikesarja tai liike on. Jokainen käyttäjä voi määrittää itse omat pisteensä, jolloin generaattoria voidaan virittää vieläkin enemmän käyttäjälle sopivaksi.

Generaattorin asetuksiin tulee mahdollisuus valita, kuinka vaikeita liikesarjoja haluaa generoida. Asetuksiin tulee myös mahdolliseksi kieltää kahden liikkeen esiintyminen peräkkäin. Useamman liikesarjalistan samanaikainen käyttö on myös toteutettavien asioiden listalla.

Generaattorin parantamisen lisäksi sivustolle on tulossa liike- ja liikesarjalistojen hallintapaneeli. Listoihin on suunnitteilla myös mahdollisuus kirjoittaa muistiinpanoja sekä merkitä, jos on onnistunut jossain liikesarjassa tai liikkeessä. Sivustolle on myös tulossa kattava kommentointijärjestelmä. Esimerkiksi liikkeen omalla sivulla voisi kysyä kommenteissa apua liikkeen suorittamiseen.

Sivuston jatkokehitys kattaa myös ohjelmointivirheiden korjaamista ja yleistä toiminnan ja sivuston käytettävyyden parantamista.

### **Ideoita**

Sivustoa voi laajentaa vielä paljon. Voisin tehdä sivustolle oman osion harjoituspäiväkirjalle. Harjoituspäiväkirjan voi yhdistää hyvin liikesarjojen ja generaattorin kanssa. Oman harjoitteluohjelman luominen ja sen seuraaminen voisi olla mahdollista. Näitä harjoitteluohjelmia voisi luoda ja jakaa muiden harrastajien kanssa.

Sivustolle voisi toteuttaa yhteisöt, jotka yhdistäisivät saman alueen triikkaajia. Omien harjoittelupaikkojen listaaminen kartalle ja niistä perustietojen kertominen, kuten maksut ja aukioloajat voisivat auttaa paikkakunnalla vierailevaa triikkaajaa. Nämä tiedot voisi myös nähdä suoraan mobiililaitteilla niin, että käyttäjän sijainti tunnustetaan, ja sen mukaan näytetään, missä on lähimmät harjoitussalit, ja ketkä siellä käyvät harjoittelemassa.

Sivustolle voisi lisätä saavutusmerkit. Kun käyttäjä suorittaa tietyn tehtävän myönnetään hänelle merkki tehtävän suorittamisesta. Erilaisia merkkejä voi keksiä vaikka kuinka paljon. Esimerkiksi 100 tallennettua liikesarjaa, 10 itse lisättyä liikettä, tai jos on lukenut ohjeet generaattorin käyttöön. Saavutusmerkit varmasti aktivoisivat käyttäjiä käyttämään Combogeneraattoria paljon enemmän ja monipuolisemmin

Combogeneraattorista voisi toteuttaa offline-version tai natiiviohjelmiä eri laitteille. Myös monikielisyys on käynyt mielessä. Siihen kyllä tarvitsen innostuneita kielitaitoisia triikkaajia avuksi.

Olen aina kirjoittanut muistiin uusia ideoita sitä mukaan, kun olen niitä keksinyt tai joku toinen on niitä minulle ehdottanut. Näitä ideoita tulee varmasti vielä lisää, ja tulen vielä karsimaan niistä osan pois. Kuitenkin tavoitteena olisi laajentaa generaattoria mahdollisimman paljon seuraavien vuosien aikana.

## 7 LOPPUSANAT

Combogeneraattori on tähän asti laajin työ, minkä olen tehnyt. Välillä tuntui siltä, että työ ei vain koskaan valmistu ja aluksi ei ollut mitään tietoa, kuinka esimerkiksi generaattorin saisi generoimaan sellaisia liikesarjoja, joita voidaan oikeasti tehdä. Sain kuitenkin kaikki asiat ratkaistua ja työ valmistui lopulta. Opin valtavan määrän uusia asioita ja tekniikoita sivuston suunnittelusta ja toteutuksesta. Työn tekeminen oli erittäin hyvä harjoitus ammatillisenosaamiseni parantamiseen.

Tämä oli erittäin hyvä lähtö Combogeneraattorille. Niin kuin johdannossa sanoin, halusin saada tämän valmiiksi ja sen takia tein tästä opinnäytetyöni aiheen. Projekti kuitenkin on niin mieluinen ja hyödyllinen, että jatkan sen kehittämistä ja lisään uusia ominaisuuksia aina sitä mukaan kun saan niitä valmiiksi.

## LÄHTEET

Bootstrap: Documentation. 2013. Luettu 21.4.2013.

<http://twitter.github.io/bootstrap/index.html>

Dive into HTML5. What Does It All Mean? 2011. Luettu 17.4.2013.

<http://diveintohtml5.info/semantics.html>

Dive into HTML5: Detecting HTML5 features. 2011. Luettu 17.4.2013.

<http://diveintohtml5.info/detect.html>

jQuery: What is jQuery? 2013. Luettu 21.4.2013. <http://jquery.com/>

LESS. 2013. Luettu 17.4.2013. Luettu 21.4.2013. <http://lesscss.org/>

MacIntyre, P. 2010. PHP: The Good Parts. O'Reilly Media, Inc.

Mozilla Developer Network. CSS 2012. Luettu 20.8.2012.

<https://developer.mozilla.org/en-US/docs/CSS>

MySQL 5.6 Manual. What is MySQL? 2013. Luettu 21.4.2013.

<http://dev.mysql.com/doc/refman/5.6/en/what-is-mysql.html>

PHP: Preface. 2012. Luettu 20.8.2012

<http://fi2.php.net/manual/en/preface.php>

Sinkkonen, I., Nuutila, E., Seppo, T. 2009. Helppokäyttöisen verkkopalvelun suunnittelu. Tietosanoma Oy.

Ullman, L. 2012. Modern JavaScript: Develop and Design. Peachpit Press.

W3C. Plan 2014 2012. Luettu 21.4.2013.

<http://dev.w3.org/html5/decision-policy/html5-2014-plan.html#html5.0-milestones>

W3C. HTML & CSS 2012. Luettu 20.8.2012.

<http://www.w3.org/standards/webdesign/htmlcss>

Winesett, J. 2010. Agile Web Application Development with Yii 1.1 and PHP5. Packt Publishing Ltd.

Wroblewski, L. 2011. Mobile First. A Book Apart.

Xue, Q., Wei Zhuo, X. 2012. The Definitive Guide to Yii 1.1.

Yii Framework: Class Reference. 2013. Luettu 21.4.2013.

<http://www.yiiframework.com/doc/api/1.1/#zii.widgets.jui>

Yii Framework: Extensions. 2013. Luettu 22.4.2013.

<http://www.yiiframework.com/extensions/>