Millionaire

Spelets idé:

Spelaren ska med hjälp av ett javaprogram kunna spela ”Vem vill bli miljonär”. Detta ska ske genom att spelaren i ett userinterface ska få frågor presenterade och klicka på rätt svar. Frågorna ska bli svårare och svårare ju längre upp för ”prisstegen” spelaren klättrar. Frågorna handlar om systemutveckling och är uppdelade i fyra delar(CSS, JAVA, HTML och Javascript).

Spelet har fyra stycken ”livlinor” som är ”50/50” där två av de felaktiga svaren plockas bort, ”byt fråga”, ”fråga publiken” och ”ring en vän”.

Om spelaren klarar alla 15 frågor har man vunnit det maximala beloppet. Spelaren kan alltid välja att stanna även om spelaren läst frågan och då erhålla det beloppet spelaren just då har tjänat in.

Svarar man fel förlorar spelaren alla pengar eller åker ner till säkerhetsnivån och får endast dessa pengar. Pengarna förs vidare med namn till databasen och ska där sorteras i en ranking som sen ska laddas upp till hemsidan. Erhåller två spelare lika mycket pengar är det den som varit snabbast som lägger sig högre upp på highscoretabellen.

För fullständiga regler läs avdelningen regler på hemsidan.

Gruppens medlemmar:

**Henrik:** Har ansvarat för dokumentation, hemsidans subsida om funktionell programmering och diverse javakodning med bl.a. livlinor, säkerhetsnivåer och userinputscreen. Skapade frågorna om Java.

**Jesse:** Har ansvarat för hemsidans indexsida och subsida om reglerna, diverse javakoding som bl.a. livlinorna, endofgamescreen och pristabellen. Skapade frågorna om Javascript.

**Joakim:** Har ansvarat för att göra grunden till hemsidan, footern på hemsidan, diverse javakodning som bl.a poänguträkningen, återställning av spelet och layout. Skapade frågorna om HTML.

**Millad:** Har ansvarat för hemsidans aboutsida och diverse javakodning som timerlayout och en del av databaskopplingen.

**Mohammad:** Har ansvarat för hemsidans subsida highscore, databashanteringen, strukturen i Javadelen och många javadelar som bl.a. frågorna, timern, ljudeffekter, intro och grafik. Har mergat med mastern på GIT. Har skapat UMLdiagram och strukturbilderna i dokumentdelen. Skapade frågorna om CSS.

Struktur på javadelen: (SKA SKRIVAS OM)

Vi har utgått från ett MVC designpattern. Dessutom har vi en del klasser som är Singleton och även använt oss av Strategypatten när vi gjort livlinorna.

Struktur på HTMLdelen: (SKA SKRIVAS OM)

Startsidan har en navigationbar längst upp med länkar till subsidorna. Själva innehållet ger en introduktion med bilder och text till spelet och subsidorna med regler, highscore och funktionell programmering. Bilder och namn på oss skapare ligger i footern längst ner.

Planerade utvecklingar:

Hade mer tid funnits hade vi gjort ett webbaserat spel. Vi hade även skapat multiplayerfunktion.

Fixa en loadingscreen när programmet hämtar datan från databasen och tillhörande errormeddelande.

Vi hade skapat olika kategorier man kunde välja som tex sport, musik, historia, geografi eller som det är på orginalspelet en blandning av allt.

Vi hade skapat en screen på slutet som i händelse av att spelaren tappat uppkoppling kan få information om detta. I nuläget avslutas spelet korrekt i avsaknad av uppkoppling men ingen highscore skickas då till databasen.

Hade vi haft mer tid hade vi även skapat en webbaserad version av spelet.

Säkerhet: //TODO

Veckovis utveckling:

Vecka 1:

Alla skapade sina frågor och laddade upp dessa på databasen som Mohammad skapade. Joakim började skissa på hemsidan och Jesse på javadelen. Henrik tog tag i organiseringen.

Vecka 2:

Millad anslöt sent till gruppen och denna vecka bestämde vi att vi skulle använda JavaFX till projektet så alla läste på om detta. Mohammad lade upp projektet på Github och alla anslöt sig till detta. Mohammad utvecklade Jesses javagrund och kopplade denna till databasen.

Vecka 3:

Denna vecka var ren javakodning där alla gjorde metoder till spelet och spelet tog form ordentligt.

Vecka 4:

Mohammad skapade alla bilder som Henrik satte på plats. Jesse fick ordning på pristabellen. Mohammad slutförde så att frågorna och svarsalternativen kom på plats. Samtidigt arbetade allihopa med metoder i Java.

Vecka 5:

Javakodningen fortlöpte när alla utom Mohammad jobbade med varsin ledtråd. Mohammad väntade in alla andra och arbetade då parallellt med en potentiell webbversion av spelet.

Vecka 6:

Mohammad hjälpte de som kört fast för att gruppen kände att det börjar bli dags att sätta spelet och att vi vill ha klart det nästa eller nästnästa vecka. Mohammad färdigställde även timern och ljudeffekterna. Henrik gjorde klart userinputscreen och Jesse endgamescreen. Millad tog tag i hur spelet skicka spelomgången till databasen.

Vi gjorde även regler för vad som ska gälla för gruppen. Inte för att det hade hänt något utan för att vi glömt det till en börja. Vi planerade även hur hemsidan ska se ut. Denna tog återigen Joakim tag i och började designa denna.

Vecka 7:

Spelet var nu spelbart men det återstod många buggfix och metoder för att få det som vi ville. Millad gjorde en grafisk detalj med timern och arbetade sen med hemsidans aboutsida. Joakim löste metoden som räknar ut playerscore och grafiska detaljer med texten. Jesse började med subsidan regler. Henrik tog tag i dokumentdelen och subsidan funktionell programmering. Mohammad fick designa om strukturen då vi upptäckt att denna behöver justeras för att få ett så bra designpattern som möjligt. Han började även på subsidan highscore och hur denna ska läsa in datan från databasen.

Vecka 8:

Alla hade arbetat med sin HTMLdel. Millad hade arbetat med aboutsidan, Henrik med funktionell programmering, Jesse med regelsidan, Joakim med footern och Mohammad med highscoresidan. Mohammad hade även strukturerat om klasserna.

Vecka 9: Alla hade under jullovet arbetat med sin del i javadoc och sin del av hemsidan. Mohammad hade gjort diverse diagram och skrivit texten som tillhör dessa. En del buggar har även fixats i spelet.

Vecka 10: Presentation!

Utvärdering av projektet:

Det vi gjorde bra var att vi klarade av det vi verkligen ville göra. Vi samarbetade till slut bra och om någon körde fast hjälpte vi varandra istället för att projektet stod still. Vi kommunicerade bra på messenger och använde Trello och GIT.

Det vi kunde gjort bättre var att redan från början tänka ut designpatterns och hur strukturen skulle vara. Detta saknade vi dock kunskap om då. En annan sak vi kunde ha gjort är att från början skriva ner alla klasser och metoder som skulle behövas men det hade varit svårt och något sätt var det denna gång bra att addera klasser och metoder allt eftersom.

GIT fungerade till slut ganska bra. Det var svårt i början men ju mer träning man fick desto bättre gick det. Man lärde sig även med tiden att skriva bra commitkommentarer och att inte committa för ofta. Då detta var första gången vi använde GIT och alla var osäkra så arbetade vi på det sättet att alla pushade upp sin egen branch och Mohammad mergade alltid dessa med Master branch. På så sätt minimerade vi konflikter i koden.

Användning av CMS i projektet?: (SKA BARA PRATAS OM PÅ DEMO)

Då vi ändå behärskar HTML, CSS och JAVASCRIPT var det lättare att inte använda CMS. CMS passar bra för personen som inte alls kan koda men ändå vill ha tex en blogg. Huruvida all funktionalitet hade gått att få genom CMS vet vi inte. Det finns ju idag väldigt avancerade och skräddarsydda CMS som en del kostar väldigt mycket pengar. Men ju mer avancerade CMSerna blir desto mer krävs det av användaren att sätta sig in i de avancerade funktionerna. Så då är det i vårat fall bättre att koda själva.

Backlog:

Så här såg våran backlog ut. Den har såklart ändrats då vi kommit på fler saker att göra och att vi till en början missuppfattat några av kraven.

BACKLOG MILLIONAIRE

Veckovis planering: Checklist med tio stycken objekt(en för varje vecka) där möte och dokumentation ska utföras.

Javadelen: Checklist med 5 objekt

Leta upp ljudfiler

Börja bygga grunden

Spara highscore för att sedan kunna lägga till i databas.

Kommentera för JavaDoc

Arv polymorfism

GUI, Swing

Versionshantering GIT

HTMLdelen:

Funktion för att läsa från databasen och presentera datan. MÅSTE använda AJAX

Funktion för att läsa reglerna och förstå hur highscorelistan konstrueras.

Del som beskriver vad funktionell programmering är.

Dynamsika webbapplikationer med serversidespråker PHP.

Versionshantering GIT

UX

Göra hemsidan responsiv

Databasen:

Skaffa db grunden.

Skaffa addera ny fråga delen (front+back)

Utveckla addera ny fråga delen

Skaffa php sidan för att hämta spel frågor.

Skaffa highscore delen (back).

Spelplanering:

Kortfattat vad spelet ska gå ut på

Komma på 15 frågor per person om varsitt ämne

Inlämning/Github:

Projektets gång

Inlämning backend

Inlämning frontend

Inlämning Worddocument

Inlämning powerpoint

Dubbelkolla kraven som Rickard gett oss

Documentdelen:

Organisera Trello

Skapa Messengergrupp

Veckovis log: Vem , vad?

Backlog(Initiala versionen)

Förklaring av spelet

Gruppmedlemmar

Ansvarsområde för varje medlem

Struktur av spel: klasser, metoder, variabler. Ska visas med Bilder och text.

Varför valde vi våra datastruktur. Arrays, list?

Varför valde vi och hur ser vårat designmönster ut? MCV?

Struktur av hemsida.

Vilken säkerhet har vi och vilken säkerhet skulle vi vilja ha?

Vilka funktioner skulle vi vilja ha om det fanns mer tid?

Utvärdering. Bra, dåligt. +-!?. GIT?

Vad är algoritmer?

Redogöra för versionshantering, GIT

Vad är CMS?

Vad är funktionell programmering?

Vad är tex XSS(Cross-site Scripting). Säkerhet i script.

Presentation:

Gå igenom dokumentets punkter

Demonstrera projektet

Fördelar/nackdelar med CMS