Design Rationale – Outsourcify

Översikt över designval

1. Layout för felrapportering förstasida
2. Rangordning av fel förstasida
3. Layout för felspecificering förstasida
4. Layout för bekfräftelse av felrapportering andrasida
5. Design för navigation till förstasida från andrasida
6. Färgval i appen
7. Lokalisering av bussen
8. Implementera FörarID eller inte
9. Implementera BussID eller inte
10. Val av minimum API
11. Val av target API
12. Utveckla för surfplatta eller mobiltelefon
13. Skapa designelement genom XML eller Java

Designval

Problem #1: Hur ska vi designa layouten för felrapporteringen på förstasidan?

Alternativ A: Androids GridView, ett rektangulärt rutnät, med stora bild-knappar och små förklarande texter.
Fördelar: Effektiv användning av yta, passar för kvadratiska bilder.
Nackdelar: Svårt att skräddarsy layout.

Alternativ B: Android's ListView, en lista med objekt, med förklarande texter.
Fördelar: Lätt att skräddarsy layout.
Nackdelar: Ineffektiv användning av yta, passar inte för kvadratiska bilder.

Alternativ C: Egen version av ett rutnät, med stora bild-knappar och små förklarande texter.
Fördelar: Effektiv användning av yta, passar för kvadratiska bilder, lättare att skräddarsy, dvs. kontrollera komponenternas design och funktionalitet.
Nackdelar: Extra arbete i samband med framtagning och underhåll.

Beslut: Vi valde alternativ A, Androids GridView. Fördelarna bedöms vara det bästa för vår applikation, då fokus ligger på en tydlig och visuell layout. Nackdelarna med GridView påverkar oss inte i någon större utsträckning, då layoutens basfunktionalitet är precis det vi letar efter. Alternativ Bs nackdelar bedöms motverka vår vision. Alternativ C bedöms inte vara mer lämpligt än alternativ A då möjligheterna till en mer skräddarsydd layout inte skulle utnyttjas. Dessutom skulle detta innebära att vi inte använder oss av Androids standard designmönster, och på så sätt utsätta oss för risken att vår version inte är kompatibel med framtida versioner av Android.

Problem #2: I vilken ordning ska felen för felrapporteringen listas på förstasidan?

Alternativ A: Hårdkodat utefter vanligast förekommande i fallande ordning, med vanligast längst upp till vänster.

Fördelar: Möjlighet till optimal snitt-användarupplevelse.
Nackdelar: Kräver uppdatering.

Alternativ B: Dynamiskt och uppkopplat utefter vanligast förekommande i fallande ordning, med vanligast längst upp till vänster.

Fördelar: Möjlighet till optimal snitt-användarupplevelse. Automatisk uppdatering av layout utefter förekomst.
Nackdelar: Förändrad layout kan orsaka förvirring.

Alternativ C: Alfabetisk ordning.

Fördelar: Vedertagen standard för uppradning. Kräver ingen uppdatering.
Nackdelar: Riskerar att användarfokus hamnar fel. Risk för förändrad layout vid översättning mellan olika språk.

Beslut: Vi valde alternativ A, hårdkodat utefter vanligast förekommande i fallande ordning, med vanligast längst upp till vänster. Detta innebär att användarna behöver lägga minst tid på att hitta rätt fel att rapportera in, givet att rangordningen är uppdaterad och att fokus efterliknar det sättet vi läser på i västvärlden. Det bör nämnas att vissa användare, med exempelvis Arabisk bakgrund, läser från höger till vänster och att rangordningen inte är optimalt för dessa. Detta bedöms dock endast utgöra ett problem i en minoritet av fallen, då vi har antagit att majoriteten av användarna är bekväma med att läsa från vänster till höger. Om det visar sig att detta antagande är felaktigt kan rangordningen behöva ses över i framtiden. Alternativ A valdes över alternativ B då genom en hårdkordad hantering erhålls en större kontroll än en dynamisk. Vidar bedömdes det att när väl layouten är invand kan kontinuerliga förändringar få en negativ effekt på förarnas användarupplevelse. Det extraarbete som uppkommer i samband med en dynamisk hantering, lämpligen via en databas, bedömdes därför vara onödigt. Alternativ C bedömdes inte vara lämpligt då den motarbetar visionen i samband med överbryggandet av språkbarriärer. Detta för att förare med olika rangordningar kan uppleva problem vid kommunikation med andra förare eller trafikledning. Alternativ C innebär även en risk för att användarfokus hamnar fel vilket kan resultera i en ineffektiv användning och en försämrad användarupplevelse. Det bör tilläggas att om situationen kring felrapporteringen skulle ändras drastiskt, så att en viss typ av fel förekommer påtagligt oftare än i dagsläget, så kan det vara motiverat att manuellt uppdatera rangordningen i koden.

Problem #3: Hur ska vi designa layouten för ytterligare specificering av fel på förstasidan?

Alternativ A: Androids AlertDialog, ett litet fönster där användaren kan göra beslut eller fylla i ytterligare information.

Fördelar: Smidig funktionalitet, design och användning.
Nackdelar: Begränsningar i layout, med t.ex. max 3 knappar.

Alternativ B: Ny view med en lista över alternativ.

Fördelar: Full kontroll över funktionalitet och design.
Nackdelar: Krångligare navigation för användaren, fler klasser, extra arbete i samband med framtagning och underhåll.

Alternativ C: Egen Dialog, där vi utökar AltertDialog-klassen med egen funktionalitet.

Fördelar: Smidig funktionalitet, design, användning och full kontroll, inga begränsningar i layout.
Nackdelar: Extra arbete i samband med framtagning och underhåll.

Beslut: Vi valde alternativ A, Androids AlertDialog. Detta alternativ möjliggör att användaren kan stanna kvar på förstasidan, samtidigt som det blir väldigt intuitivt att det krävs ytterligare information från användaren innan felrapporteringen kan gå vidare till nästa steg. Funktionaliteten är tillräcklig och alternativ A bedöms således vara mer lämpligt än alternativ C, då möjligheten till en mer skräddarsydd layout inte skulle utnyttjas och extraarbetet då skulle vara onödigt. Alternativ A ansågs vidare vara mer lämpligt än alternativ B, en ny view, då nackdelarna med denna bedöms motverka vår vision. Fokus läggs på tydlighet, och flera vyer riskerar förvirra användaren och krångla till flödet i felrapporteringsprocessen. Vidare anses alternativ A vara mer lämpligt än alternativ B då fördelarna med alternativ B inte skulle utnyttjas fullt ut. Det extra arbetet i samband med framtagning och underhåll skulle därför vara onödigt.

Problem #4: Hur ska vi designa layouten för status av felrapporteringen på bekräftelsevyn?

Alternativ A: Tydlig visuell bekräftelsebild med inrapporterad information och handlingsalternativ i textform under.

> **Fördelar**: Användaren kan säkerställa att rätt information rapporterats in, samtidigt som ytterligare handlingsuppmaningar inte riskerar misstolkas.
> **Nackdelar**: Ställer krav på att användarens språkkunskaper.

Alternativ B: Tydlig visuell bekräftelsebild med bild av inrapporterat problem under, tillsammans med bild över handlingsalternativ.

> **Fördelar**: Användaren kan säkerställa att rätt problem rapporterats in samtidigt som inga krav ställs på användarens språkkunskaper.
> **Nackdelar**: Användaren kan inte säkerställa att rätt felspecifiering rapporterats in, samtidigt som det finns utrymme för misstolkning av handlingsalternativ.

Beslut: Vi valde alternativ A, tydlig visuell bekräftelsebild med inrapporterad information och handlingsalternativ i textform under. Här uppstår en trade-off mellan text och bilder. Vi vill i största möjliga utsträckning minimera användandet av text, i enlighet med vår vision. Det är även av yttersta vikt att handlingsalternativen inte misstolkas, då detta kan få allvarliga konsekvenser. Det bedömdes att handlingsalternativen skulle vara för svåra att presentera i bildform på ett tillräckligt sätt, och att risken för misstolkning motiverar användandet av text i detta fall.

Problem #5: Hur ska vi designa navigation till föregående sida från bekräftelsevyn?

Alternativ A: Centrerad tillbaka-knapp placerad under all information.

Fördelar: Undviker att användarens fokus hamnar fel. Intuitiv placering.

Nackdelar: Tar mycket plats.

Alternativ B: Centrerad tillbaka-knapp placerad under all information, tillsammans med en ”navigations-header” placerad längst upp till höger, med knappar för omdirigering till förstavyn och föregående vy.

Fördelar: Möjliggör mer exakt navigering.

Nackdelar: Tar mycket plats. Flera knappar som gör samma sak. Riskerar att användarfokus hamnar fel. Extra arbete i samband med framtagning och underhåll.

Beslut: Vi valde alternativ A, centrerad tillbaka-knapp placerad under all information. Det bedömdes vara lämpligt att användarens fokus hamnar på tillbaka-knappen efter det att felrapporterings-informationen granskats. Vidare bedömdes den mer exakta navigeringen som kunnat möjliggöras genom alternativ B vara överflödig i dagsläget. Det extra arbete som alternativ B innebär bedöms därför vara onödigt. Om applikationen får flera vyer där det är motiverat att användaren skulle kunna vilja gå tillbaka och ändra eller se över något i tidigare steg kan detta beslut behöva ses över.

Problem #6: Vilka färger ska vi använda i appen?

Alternativ A: Vit bakgrund med svart text och svarta objekt.
Fördelar: Tydligt, enhetligt och hög kontrast. Lämpligt för visuella funktionsnedsättningar.
Nackdelar: Tråkigt, ingen varumärkes-koppling.

Alternativ B: Grönt, svart, grått och vitt med de färgkoder som finns på Electricitys hemsida.
Fördelar: Varumärkes-koppling, mer visuellt attraktivt, inte tråkigt.
Nackdelar: Extra arbete, inte lämpligt för visuella funktionsnedsättningar.

Beslut: Vi valde alternativ A, vit bakgrund med svart text och svarta objekt. Stöd för funktionsnedsättningar gynnas med detta alternativ, då låga kontraster och färgblindhet kan orsaka problem med alternativ B. Detta är i enlighet med vår vision, då anvädbarhet vinner över graden av visuell attraktion. En varumärkeskoppling och en mer visuellt attraktiv färgsättning bedöms inte tillföra något värde, och det extra arbete som uppkommer i samband med alternativ B bedöms därför vara onödigt.

Problem #7: Hur ska vi hämta in bussens position och i vilket format?

Alternativ A: Enhetens GPS, koordinater.
Fördelar: Inga beroenden av externa system, pålitligt, enkelt.
Nackdelar: Högre energiförbrukning, tar inte hänsyn till existerande lösningar.

Alternativ B: Extern API, koordinater, gatunamn, busshållplats.
Fördelar: Möjlighet till bättre passform med existerande system.
Nackdelar: Förlitar sig på externa system, högre komplexitet.

Beslut: Vi valde alternativ A, enhetens GPS. Då vi inte har någon insyn i de externa systemen i samband med Electricitys API:er bedömdes enhetens inbyggda GPS funktionalitet vara ett mer pålitligt val vad gäller ”up-time”. Den högre energiförbrukningen på enheten accepteras då enheten förväntas vara på laddning under färd. Även användningen av koordinater bedöms utgöra en tillräckligt bra passform med mottagarens system, då denna rimligtvis någon gång hanterar koordinater i sitt system och kan på så sätt även konvertera våra medskickade koordinater till önskat format, såsom gatunamn eller busshållplats.

Problem #8: Ska vi implementera användningen av ID för förare, i så fall hur?

Alternativ A: Ja, genom extern API.

Fördelar: Stämmer överens med utskick av data. Ger möjlighet att på ett smidigt sätt sammankoppla förare med felrapporteringar.
Nackdelar: Förlitar sig på externa system, högre komplexitet.

Alternativ B: Ja, genom manuell inmatning.

Fördelar: Stämmer överens med utskick av data. Ger möjlighet att på ett smidigt sätt sammankoppla förare med felrapporteringar.
Nackdelar: Risk för missbruk, utebliven användning och mer komplicerad användarupplevelse.

Alternativ C: Nej.

Fördelar: Möjlighet till användning av mottagarens system.
Nackdelar: Risk att mottagarens system inte erbjuder funktionalitet för sammankoppling av förare med inrapporterat fel.

Beslut: Vi valde alternativ C, att inte implementera användningen av ID för förare. Detta förutsätter att mottagarens system kan länka samman förare med inrapporterat fel genom den övriga information applikation skickar, dvs. koordinater och tidpunkt. Denna förutsättning bedöms vara trolig, då mottagarens system i dagsläget använder scheman och rutter. Alternativ A och B hade varit att föredra om respektive nackdelar inte varit så påtagande. Vad gäller alternativ A kan detta behöva ses över om API:erna skulle bli mer pålitliga i framtiden. Vad gäller alternativ B skulle detta innebära en mer utdragen rapporteringsprocess med risk för att fel uppgifter rapporteras. Detta går emot vår vision, där kvalitet på informationen och en smidig process efterfrågas.

Problem #9: Ska vi implementera användningen av ID för bussar, i så fall hur?

Alternativ A: Ja, genom extern API.

Fördelar: Stämmer överens med utskick av data. Ger möjlighet att på ett smidigt sätt sammankoppla bussar med felrapporteringar.
Nackdelar: Förlitar sig på externa system, högre komplexitet.

Alternativ B: Ja, genom manuell inmatning.

Fördelar: Stämmer överens med utskick av data. Ger möjlighet att på ett smidigt sätt sammankoppla bussar med felrapporteringar.
Nackdelar: Risk för missbruk, utebliven användning och mer komplicerad användarupplevelse.

Alternativ C: Nej.

Fördelar: Möjlighet till användning av mottagarens system.
Nackdelar: Risk att mottagarens system inte erbjuder funktionalitet för sammankoppling av bussar med inrapporterat fel.

Beslut: Vi valde alternativ C, att inte implementera användningen av ID för bussar. Detta förutsätter att mottagarens system kan länka samman bussar med inrapporterat fel genom den övriga information applikation skickar, dvs. koordinater och tidpunkt. Denna förutsättning bedöms vara trolig, då mottagarens system i dagsläget använder scheman och rutter. Alternativ A och B hade varit att föredra om respektive nackdelar inte varit så påtagande. Vad gäller alternativ A kan detta behöva ses över om API:erna skulle bli mer pålitliga i framtiden. Vad gäller alternativ B skulle detta innebära en mer utdragen rapporteringsprocess med risk för att fel uppgifter rapporteras. Detta går emot vår vision, där kvalitet på informationen och en smidig process efterfrågas.

Problem #10: Vilken minimum API-level ska vi tillåta i applikationen?

Alternativ A: 16, Jelly Bean.

Fördelar: Den version som vi har på vår surfplatta. Stödjer all funktionalitet i nuvarande version av appen. Täcker in 95.7% av alla enheter i dagsläget.
Nackdelar: Kanske inte stödjer efterfrågad funktionalitet i framtiden.

Alternativ B: 23, Marshmallow.

Fördelar: Stödjer den senaste funtionaliteten inom Android.
Nackdelar: Täcker endast in 7.5% av alla enheter i dagsläget

Alternativ C: 14, Ice Cream Sandwich.

Fördelar: Stödjer all funktionaliet i nuvarande version av appen. Täcker in 97.7% av alla enheter i dagsläget.
Nackdelar: Kanske inte stödjer efterfrågad funktionalitet i framtiden.

Beslut: Vi valde alternativ C, Ice Cream Sandwich. Genom att använda en tillräckligt låg API, som i dagsläget täcker in 97.7% av användarbasen för Android-enheter, kan vi säga att det med största sannolikhet inte kommer bli några problem vid ett potentiellt införande av produkten. API 14 stöjder all funktionalitet som appen efterfrågar i dagsläget. Om detta skulle ändras i framtiden bedöms problemen som uppstår inte vara allvarliga. Potentiella lösningar kan vara att ta fram två olika versioner av appen, en med högre API som stödjer ny funktionalitet och en gammal version som fortfarande fungerar på äldre system. En annan aspekt som bör finnas i åtanke är att användarbasen av Android-enheter kontinuerligt börjar använda högre API:er. Det kan mycket väl vara så att användarbasen redan använder en högre API än den minimum-API vi uppdaterar till vid uppdateringstillfället.
(https://developer.android.com/about/dashboards/index.html)

Problem #11: Vilken target API-level ska vi använda i applikationen?

Alternativ A: 16, Jelly Bean.

Fördelar: Den version som vi har på vår surfplatta.
Nackdelar: Existerande funktionalitet kanske inte lever kvar i nya högre API-versioner av Android.

Alternativ B: 23, Marshmallow.

Fördelar: Den senaste versionen av Android.
Nackdelar: Går inte att testa på vår surfplatta med API 16.

Beslut: Vi valde alternativ B, Marshmallow. Detta för att det bedömdes lämpligt att säkerställa att applikationen fungerar som tänkt även på de högre nivåerna av API-versioner. Användarbasen av Android-enheter börjar kontinuerligt använda högre API:er. Det finns ingen anledning att inte säkerställa att enheten fortfarande kommer att fungera efter det att majoriteten av Android-användarbasen har installerat Marshmallow. Nackdelen, dvs. att v inte kan testa det på vår surfplatta, bedöms vara försumbar då vi kan kompensera med tester i emulerade android-enheter med API 23.

Problem #12: Vilken typ av enhet ska vi utveckla till?

Alternativ A: Mobiltelefon.
Fördelar: Flest enheter.
Nackdelar: Liten skärm.

Alternativ B: Surfplatta.
Fördelar: Storm skärm.
Nackdelar: Färre enheter.

Beslut: Vi valde alternativ B, sufplatta. Vi valde att utveckla till surfplatta då detta bäst stämmer överens med vår vision kring att främja användarvänligheten. Ett större skärm möjliggör större ikoner och en mer tydlig layout visuellt. Nackdelarna med antalet enheter bedöms inte påverka oss i någon större utsträckning, då vår användarbas är väldigt centrerad och att användningen av appen till stor det kan komma att styras av företaget som förarna jobbar hos. Om appen tas i bruk är det alltså lämpligt att en surfplatta installeras i samtliga bussar där appen ämnar användas.

Problem #13: Hur ska vi skapa designelement kodmässigt?

Alternativ A: I Java-klasserna.

Fördelar: Intuitivt. Vedertagen standard i samband med kodutveckling i Java. Gruppen besitter kunskaper inom detta sen tidigare.

Nackdelar: Använder inte Androids standard designmönster

Alternativ B: Genom XML.

Fördelar: Androids standard designmönster. Vedertagen standard i samband med kodutveckling i Android.

Nackdelar: Gruppen besitter inte kunskaper inom detta sen tidigare.

Beslut: Vi valde alternativ B, genom XML. Detta för att användningen av Androids standard designmönster bedömdes vara bättre än att använda sig av designmönster i samband med Java, kring just detta. Efter inlärning av det nya sättet att utveckla och använda design-element på föväntas även XML fungera smidigare eftersom det är framtaget och anpassat efter just denna nisch av kodutveckling. Det extraarbete som uppkommer i samband med inlärning bedöms därför vara motiverat.