# Proposal for “Plan A” UK-scale habitat change detection application

General technical goals

* Enable incremental processing of more areas, to full UK-scale
* Support potential alternative detection methods/algorithms
* Make the app 10-1000x faster
* Make thumbnails appear 100-10000x faster
* Modern user-focussed UI/UX
* Save state between sessions
* Don’t time-out
* Show more information about each spatial framework / habitat map

## Technology stack

JNCC are fortunate to have available a combination of technical skills and experiences that are highly-aligned with the development challenges of this application.

* We have experience in supercomputing JASMIN, Unix
* We use AWS cloud services for our day-to-day application hosting
* We have recently designed and built a modern web app for Scottish Government where some of the front-end challenges are similar

## Architecture diagram

JASMIN AWS  
  
|---------------| |---StatsService---|  
| | stats | |   
| stats, thumbs | -------> | Postgres |  
| sqllite | | | |  
| | | |  
|groupworkspace | |------------------|  
|---------------| | API |  
 \ |------------------|  
 \ |   
 thumbs \ | json  
 \ V  
   
 UI web browser app :-)

## Offline processing (Jasmin)

The fundamental indexes and statistics will be calculated from analysis-ready data on the Jasmin supercomputer.

The inputs are:

* A spatial framework (a habitat map)
* Sentinel 1 and Sentinel 2 analysis-ready data

The outputs are:

* a database of polygon statistics; that is statistics about indexes for each polygon in the habitat map
* thumbnail images for each row

The outputs are placed on the public Internet-accessible “group workspace”. From there, the statistics database will be shipped somehow to an AWS Postgres database. The thumbnail images can be requested directly by the app running in a user’s browser.

* Jasmin
* Python, R, Luigi

## Back-end (AWS)

The app will be retrieve its data from a Postgres relational database containing the statistics table, spatial framework information and the gazetteer database.

The app will be powered by “serverless” HTTP/JSON API which sits beteen the database and the Internet.

* AWS cloud services
* Postgres (AWS RDS)
  + Our go-to RDBMS with spatial support
  + Considering Aurura when v2 supports Postgres
* Cloud functions (AWS Lambda)
* Cloud object storage (AWS S3)

## Front-end

We will build on the successful development of the Scottish Remote Sensing Portal https://remotesensingdata.gov.scot/.

* React + Redux. React from Facebook is one of the leading frameworks for rich web-based user interfaces. Manage application state, user application session storage.
* Typescript. A programming language developed and maintained by Microsoft. It is a strict syntactical superset of JavaScript and adds optional static typing to the language. TypeScript is designed for the development of large applications and transpiles to JavaScript.
* Leaflet. A JavaScript library for interactive maps.
* Bootstrap + JNCC house stylesheets, Sass.

## Open source at JNCC

* All proposed technologies are open source (with the notable exception of AWS cloud services).
* We will build the system in the open, on http://github.com/jncc, under an open licence (OGL? #TODO).

## Risks and challenges

* We are not yet able to be confident that the Plan A architecture proposal will scale. We need to have the Yorkshire stats

## Further thoughts

* Potentially enable hires images / link to source ARD