# Implementatieplan voor de efficiëntste container

## Namen en datum

Florian Humblot en Vera Schoonderwoerd

Datum: 11-02-19

## Doel

Het doel van de implementatie is om een container te bouwen voor RGB en Grayscale images met een zo snel mogelijke access time.

## Methoden

Je geeft hier aan welke methoden er zijn, wat de verschillende tussen de methodes zijn.

* Arrays
  + Pros
    - Access complexiteit is O(1) waardoor snelheid voor getPixel vrij hoog is
    - De array mag fixed-size zijn, aangezien de width en height van het plaatje ofwel bij de constructor bekend wordt gemaakt of bij de set functie. Daardoor kunnen we een array of de heap maken en in een keer de goeie size hebben.
  + Cons
    - Fixed size – uitbreiding is costly
    - Insertion en deletion time is relatief traag (O(n))
* Vectors
  + Pros
    - Access complexity is O(1)
    - Mocht de size veranderen hoeft niet de hele vector opnieuw gegenereerd te worden.
  + Cons
    - Overhead
* Linked lists
  + Pros
    - Neemt weinig ruimte in
    - Uitbreidbaar zonder de hele list de herbouwen
  + Cons
    - Access time is hoog

## Keuze

Je geeft een onderbouwing over waarom een bepaalde methode is gekozen, en/of waarom bepaalde settings zijn gebruikt.

Wij hebben uiteindelijk gekozen voor een 2 dimensionale vector, Dit omdat de overhead van een vector tegenover een array minimaal is en vectors makkelijker te beheren zijn dan arrays. De linked lists hebben wij niet gekozen omdat wij voornamelijk bezig zullen zijn met access en niet met het maken van de structuur, waarbij het voor linked lists moeilijker is om een bepaalde pixel op te vragen omdat die structuur geen random access biedt.

## Implementatie

Je geeft aan hoe deze keuze wordt geimplementeerd in de code

Wij gaan de RGBImageStudent classe implementeren met een vector van vectors van RGB pixels. Waarbij de buitenste vector de x-as representeert en de binnenste vectors de y-as, dit om ervoor te zorgen dat je de pixels kunt benaderen met `pixels[x][y]`. Als er een aanvraag wordt gedaan voor een pixel die buiten de range valt (met setPixel of met getPixel) dan wordt er een exception gethrowd.

## Evaluatie

Je geeft aan welke experimenten er gedaan zullen worden om de implementatie te testen en te ‘bewijzen’ dat de implementatie daadwerkelijk correct werkt. Dit geeft direct informatie over de meetrapporten die er zullen worden gemaakt.